Universidade de Aveiro
Departamento de Electrónica, Telecomunicações e Informática,
2019

Rui Eduardo Marques Guedes

Cerca virtual inteligente para pequenos ruminantes

Smart virtual fence for small ruminants

Rui Eduardo
Marques Guedes

Cerca virtual inteligente para pequenos ruminantes
Smart virtual fence for small ruminants

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Eletrónica e Telecomunicações, realizada sob a orientação científica do professor Doutor Pedro Alexandre Sousa Gonçalves da Escola Superior de Tecnologia e Gestão de Águeda, e do Professor Doutor Paulo Bacelar Reis Pedreiras, do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro.

Este trabalho é financiado pelo Fundo Europeu de Desenvolvimento Regional (FEDER), através do Programa Operacional Competitividade e Internacionalização(COMPETE2020) do Portugal 2020. [Projeto SHEEPIT com nº 017640(POCI-01-0247-FEDER-017640)]

Cofinanciado por:

COMPETE 2020 PORTUGAL 2020 UNIÃO EUROPEIA Fundo Europeu de Desenvolvimento Regional

o júri / the jury

presidente / president	**Professor Doutor Alexandre Manuel Moutela Nunes da Mota,** Professor Associado, Universidade de Aveiro
vogais / examiners committee	**Professor Doutor Pedro Alexandre de Sousa Gonçalves** Professor Adjunto, Universidade de Aveiro (orientador)
	Professor Doutor Frederico Miguel do Céu Marques dos Santos Professor Adjunto, Instituto Superior de Engenharia de Coimbra

Agradecimentos

Quero agradecer a todas as pessoas que me ajudaram ao longo do percurso académico e, especialmente, desta dissertação. Um agradecimento especial para os meus orientadores, professor Pedro Gonçalves e professor Paulo Pedreiras, pelo apoio, dedicação e disponibilidade no desenvolvimento deste trabalho.

Quero agradecer também ao Eng. Luís Miguel Nóbrega pelo acompanhamento, disponibilidade e paciência para me ajudar com todas as dúvidas. Também um agradecimento ao José Pereira por toda a ajuda que me deu. Um abraço ao Dinis Falcão que, enquanto fazia a sua dissertação, também se mostrou sempre disponível a ajudar.

Quero também agradecer a toda minha família pelo apoio e, em especial, aos meus pais e ao meu irmão, que sem eles não teria sido possível.

Por fim, queria também agradecer a todos os meus amigos pela ajuda, motivação e por proporcionarem momentos de descompressão e lazer.

Resumo

O aparecimento de espécies vegetais indesejadas em propriedades agrícolas, mais em específico em propriedades vinícolas, é um problema comum nos dias de hoje. A remoção destas é normalmente feita com recurso a máquinas agrícolas ou herbicidas, o que não só cria problemas ambientais relacionados com a queima de combustíveis fósseis, com a poluição dos aquíferos, e com a erosão dos solos, mas também problemas relacionados com a saúde humana. Uma alternativa mais ecológica é o uso de gado para a remoção deste tipo de espécies vegetais.

O projeto SheepIT tem como objetivo desenvolver uma solução de controlo de pastagem de gado caprino para remoção de espécies vegetais invasoras em vinhas. Para isso foi desenvolvida uma coleira com sensores e atuadores capaz de monitorizar e controlar o comportamento dos animais. Estas comunicam com nós fixos (faróis) e enviam para estes informações periodicamente que são encaminhadas para um nó consumidor (gateway). Aqui é feito um processamento da informação recebida, a qual é posteriormente enviada para uma plataforma computacional onde também é tratada e disponibilizada ao utilizador.

No âmbito desta dissertação foi desenvolvido um sistema de localização que determina a posição relativa de cada coleira em relação aos faróis. A localização absoluta destes é determinada por GPS, permitindo assim obter, de forma indirecta, a localização absoluta das coleiras. Para melhorar esta localização foram desenvolvidos mecanismos para aumentar a precisão e exactidão das distâncias estimadas que são usadas na trilateração, através de fusão sensorial. Para avaliar estes mecanismos foram comparados os resultados de localização obtidos por estes mecanismos com os resultados de um módulo GPS, tendo sido obtidos erros de localização abaixo de 18m.

Abstract

The appearance of unwanted plant species on agricultural properties, more specifically on wine estates, is a common problem today. The removal of these is usually done using agricultural machines or herbicides, which can raise environmental issues. A more ecologic alternative is the use of cattle to remove this type of plant species.

The SheepIT project aims to develop a goat grazing control solution for the removal of invasive plant species in vineyards. For this purpose, a collar was developed with sensors and actuators capable of monitoring and controlling the animals' behavior. These communicate with the fixed nodes (beacons) and send this information periodically until it reaches the final node (gateway). Here the information received is processed and then sent to a computer platform where it is further processed and made available to the user.

Within the scope of this dissertation, it was developed a location system that determines the relative position of each collar with respect to the beacons. The absolute location of the beacons is determined by GPS, thus allwoing to obtain, indirectly, the absolute position of the collars. To improve the location, mechanisms have been developed to improve the accuracy of the estimated distances that are used in trilateration through sensory fusion. To assess the improvemets obtained with this technique, the obtained results were compared with the ones obtained with a GPS module. The obtained localization errors are below 18m, in realistic scenarios.

Conteúdo

Lista de Figuras

Lista de Tabelas

Siglas

μC Micro-ciclo.

AMQP Advanced Message Queuing Protocol.

AoA Angle of Arrival.

BLE Bluetooth Low Energy.

CIA Circle Intersection Algorithm.

CSV Comma Separated Values.

FED Filtro de Estados da Distância.

FER Filtro de Estados do RSSI.

FIFO First In, First Out.

FK Filtro de Kalman.

GNSS Global Navigation Satellite System.

GPS Global Positioning System.

GSM Global System for Mobile Communications.

IoT Internet of Things.

JSON JavaScript Object Notation.

LIA Line Intersection Algorithm.

MC Macro-ciclo.

PTL probability torus localization.

RSS Received Signal Strength.

RSSI Received Signal Strength Indication.

RToF Round-Trip Time of Flight.

SBL Sequence-Based localization.

SW Synchronization Window.

TAW Turn-around Window.

TDMA Time Division Multiple Access.

TDoA Time Difference of Arrival.

ToA Time of Arrival.

WSN Wireless Sensor Network.

Capítulo 1

Introdução

1.1 Motivação

O aparecimento de espécies vegetais indesejadas em terrenos de cultivo é um problema muito comum na agricultura. A solução para este problema passa frequentemente pelo uso de máquinas ou produtos químicos que, apesar de eficazes, têm algumas desvantagens. No caso do uso de máquinas este método pode levar a que das espécies não invasoras sejam danificadas. Quanto ao uso de produtos químicos, como os herbicidas, podem-se levantar questões quanto à contaminação do solo e à saúde pública. Para além disso estes métodos não produzem um resultado definitivo, o que implica que o processo tenha de ser repetido e, agravando assim os efeitos negativos descritos anteriormente e tornando o processo mais dispendiosos.

Uma alternativa às soluções apresentadas anteriormente para a remoção destas espécies vegetais passa pela utilização de animais. Este método é mais ecológico que os anteriores, visto que não há qualquer tipo de poluição, e apresenta a vantagem de fertilização do solo pela parte dos animais. Contudo esta solução necessita de um sistema de controlo e monitorização dos animais. Este trabalho de supervisionamento é feito, por norma, por pastores, mas com o passar dos anos esta profissão começou a ser exercida por um número reduzido de pessoas. Assim sendo é necessário arranjar outras soluções para o supervisionamento dos animais.

O projeto SheepIT [1] tem assim como objetivo o desenvolvimento de uma solução para o controlo de pastagem de gado caprino em terrenos vinícolas. Esta solução deve ser capaz de monitorizar e condicionar o comportamento e localização dos animais. Normalmente, o condicionamento da localização animal é feita com recurso a cercas físicas que, apesar de serem bastante eficientes, possuem algumas restrições relativamente ao custo, à sua durabilidade e ao esforço necessário quando se pretende mover o gado para outro local de pastagem. Assim surgiu, o conceito de cerca virtual, que consiste em monitorizar a localização do animal e na aplicação de estímulos quando este se aproxima a um limite, não físico, da área permitida de pastagem.

Este trabalho pretende contribuir para a melhoria do sistema de localização do projeto SheepIT.

1.2 Objetivos

Esta dissertação tem como objetivo melhorar o protótipo de cerca virtual já existente no projeto SheepIT, de modo a ser desenvolvido um sistema de baixo consumo e custo capaz de

localizar os animais e delimitar a sua área de pastagem. O sistema deve ser capaz de tirar partido dos sensores existentes, de forma a poder obter-se um nível de confiança elevado em relação à posição do animal, ou seja, deve ser munido de algoritmos avançados de decisão, como *sensor fusion*, onde os seus *inputs* são os dados fornecidos pelos sensores. De seguida é apresentada uma descrição faseada do trabalho:

- Estudo dos requisitos do sistema de localização do SheepIT;
- Estudo do sistema de monitorização e condicionamento de localização atualmente existente;
- Melhoramento dos subsistemas de sensorização e atuação atualmente existentes;
- Estudo de metodologias de *sensor fusion* e respetiva implementação;
- Testes, verificação e validação dos mecanismos desenvolvidos.

1.3 Estrutura

Após uma descrição geral deste trabalho, o resto do documento está dividido da seguinte forma:

- **Capítulo 2: Projeto SheepIT** - Neste capítulo é feita uma descrição detalhada do projeto SheepIT e da sua arquitetura.
- **Capítulo 3: Estado de Arte** - Neste capítulo são apresentados alguns sistemas de localização animal e da sua monitorização, assim como alguma tecnologia e métodos existentes para a localização. O capítulo é concluído com uma descrição do sistema de localização já existente no projeto SheepIT.
- **Capítulo 4: Arquitetura e Implementação** - Neste capítulo é apresentada a arquitetura e os mecanismos implementados no âmbito da solução proposta nesta dissertação.
- **Capítulo 5: Testes e verificação** - Este capítulo é dedicado aos resultados da solução proposta no capítulo anterior.
- **Capítulo 6: Conclusão** - Por fim, este capítulo é dedicado às conclusões do trabalho apresentado, sendo também apresentadas algumas sugestões para trabalho futuro.

Capítulo 2

Projeto SheepIT

Neste capitulo é apresentada uma descrição do projeto SheepIT, nomeadamente da sua arquitetura e dos elementos que a constituem.

2.1 Arquitetura geral

Como foi referido anteriormente, o projeto SheepIT tem como principal motivação o uso de animais para a remoção de espécies vegetais invasoras, mais especificamente, o uso de gado caprino em vinhas. Contudo, este método não pode ser usado todo o ano pois, na época de produção de fruto, os animais tendem a se alimentar dos ramos inferiores das plantas e dos frutos, afetando assim a produção.

Este projeto tem como objetivo desenvolver um sistema baseado em Internet of Things (IoT) capaz de controlar a postura dos animais, nomeadamente limitar a sua capacidade de acesso a galhos e frutos de videiras, e de implementar cercas virtuais para controlar as suas áreas de alimentação. Para além disso, o sistema deve ser também capaz de colectar e guardar dados em relação a cada animal. Tendo isto em conta, os requisitos gerais do sistema podem ser resumidos da seguinte forma: [2]:

- **Tamanho, peso e autonomia:** Uma vez que os animais de foco deste projeto são de pequeno porte, os dispositivos usados por estes devem ser de pequenas dimensões, de modo a que não cause desconforto. Além disso, devem ter uma autonomia no mínimo de 4 meses.

- **Deteção de postura:** Visto que o objetivo é usar animais em vinhas para a remoção de ervas, estes precisam de ser controlados de modo a não danificarem o produto. Portanto, o sistema deve ser capaz de detetar a postura do animal, mais especificamente, a posição da cabeça e do pescoço.

- **Cerca virtual:** O sistema deve ser capaz de controlar as áreas de pastagem. Deve ser capaz de calcular a localização dos animais com um erro próximo do erro médio de um GPS.

- **Processamento local:** O sistema deve ser capaz de processar os dados obtidos pelos sensores e processá-los, utilizando algoritmos de decisão, para a aplicação ou não de estímulos.

- **Coleta de dados:** O sistema deverá permitir que sejam coletados, processados e guardados os dados de cada um dos animais, para que seja possível fazer-se uma supervisão tanto da sua saúde como do seu comportamento.

A figura 2.1 ilustra a arquitetura geral do sistema SheepIT. Este sistema é composto por três segmentos principais: a rede local, composta por dispositivos responsáveis pela geração dos dados; a plataforma computacional, responsável pelo armazenamento e processamento dos dados obtidos pela rede local; e a interface de utilizador, responsável pela visualização dos dados processados.

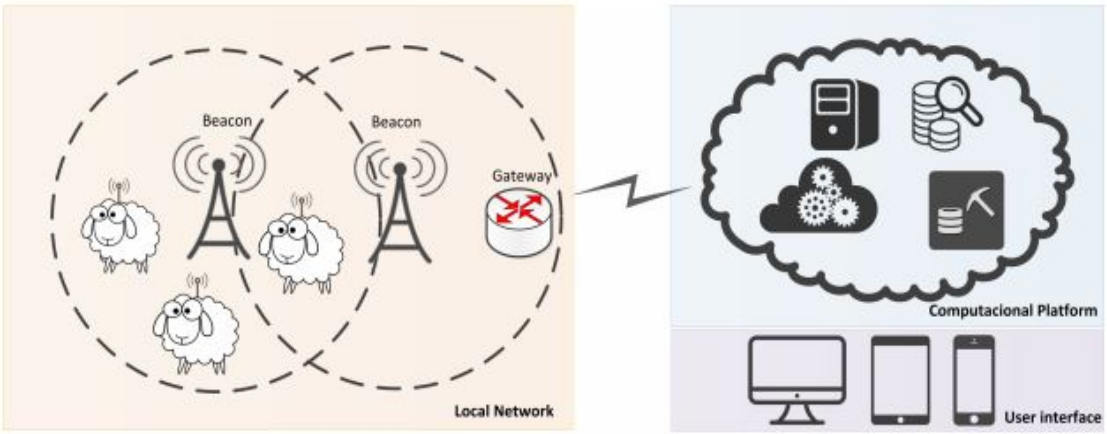

Figura 2.1: Arquitectura geral do sistema SheepIT [3]

2.2 Rede de sensores

Como é possível observar na figura 2.1 o sistema apresenta uma rede local. A rede local incorpora uma Wireless Sensor Network (WSN) que é composta por três nós: as coleiras, os faróis e a gateway. A descrição de cada um destes elementos é feita abaixo, sendo dado mais ênfase à gateway visto que o trabalho feito nesta dissertação incide sobre este elemento da rede. Além disso, é feita também uma descrição do sistema de comunicação da rede.

2.2.1 Coleiras

As coleiras são o único tipo de nó móvel presente na rede. Estas são compostas por um conjunto de sensores e atuadores, responsáveis pela monitorização e controlo da postura dos animais. O conjunto de sensores é composto por um acelerómetro, um magnetómetro e um transdutor de ultrassons, e o conjunto de atuadores é composto por um transdutor sonoro e um módulo de descarga elétrica. Para além disto, as coleiras também contêm um microcontrolador que é responsável pelo processamento dos dados obtidos pelos sensores e pela decisão da aplicação de estímulos, e um módulo de rádio responsável pelas comunicações entre as coleiras e os faróis. O *hardware* e o *firmware* desenvolvidos para as coleiras são descritos com mais detalhe em [4] e [9].

Além da postura é também necessário determinar a localização de cada animal. Por isso os autores do projeto propõem que seja determinada a sua posição relativa recorrendo ao

Received Signal Strength Indication (RSSI) presente nas comunicações entre as coleiras e os faróis. Este método permite que seja possível localizar os animais de forma barata e sem que seja preciso hardware adicional, pois o valor de RSSI é obtido através do rádio usado para as comunicações. Além do mais, este método também aumenta a autonomia, em comparação com uma localização por GPS. A figura 2.2 ilustra o protótipo de uma coleira.

Figura 2.2: Protótipo da coleira do SheepIT

2.2.2 Faróis

Os faróis, em conjunto com a gateway, constituem o conjunto de nós fixos presentes na rede. Estes elementos da rede, para além de terem como propósito o reencaminhamento das informações fornecidas pelas coleiras até à gateway, são também responsáveis pela sincronização da rede de sensores. Adicionalmente, sendo os faróis nós fixos da rede, eles são responsáveis por definir a zona de pastagem. Apesar de estes serem nós fixos, o seu posicionamento pode ser facilmente alterado de acordo com a zona que se pretende controlar.

Estes elementos são constituídos por um rádio, um microcontrolador, uma bateria e um módulo de localização. Uma vez que que estes não são transportados pelos animais podem conter uma bateria maior, o que possibilita a utilização de um módulo GPS para determinar a sua localização. Esta localização absoluta combinada com a localização relativa das coleiras, permite que seja calculada uma localização absoluta das ovelhas. A figura 2.3 representa o protótipo de um farol.

Figura 2.3: Protótipo do farol do SheepIT

2.2.3 Gateway

A gateway representa o nó final da rede. Este elemento comporta-se como uma interface entre a rede e a plataforma computacional, sendo o responsável por reencaminhar todos os dados provenientes das coleiras para a *cloud*. Este elemento desempenha um papel importante na rede pois, como todos os dados são encaminhados pelos faróis até este nó, apenas este nó tem o conhecimento do estado global da rede.

A gateway é composta por um farol ligado a um computador com conexão à Internet. Assim sendo, este elemento possui todas as funcionalidades que um farol possui, e, como é composto por um computador, é capaz de executar outras tarefas tais como: anunciar a entrada e saída de nós na rede; processar os dados das coleiras, como por exemplo, calcular a sua posição; enviar os dados recebidos e processados para a *cloud* de modo a que sejam posteriormente processados e armazenados para poderem ser acedidos remotamente; e gerar alarmes para situações anormais. O envio dos dados para a *cloud* é feito sempre que possível, sendo que no caso de não haver conexão a Internet a gateway guarda os dados para posterior envio quando a conexão for estabelecida novamente. A figura 2.4 ilustra o protótipo da gateway.

Figura 2.4: Protótipo da gateway do SheepIT

Estas funcionalidades estão atualmente implementadas em Linux, permitindo que o computador ao qual o farol está ligado, seja um PC ou *embedded* PC, como por exemplo RaspberryPi ou OrangePi0. Inicialmente estas funcionalidades eram realizadas praticamente de forma sequencial, apresentando algumas limitações a nível de eficiência de operação. Assim, é proposto um esquema de multi-thread de modo a que cada tarefa seja executada com uma taxa de execução distinta.

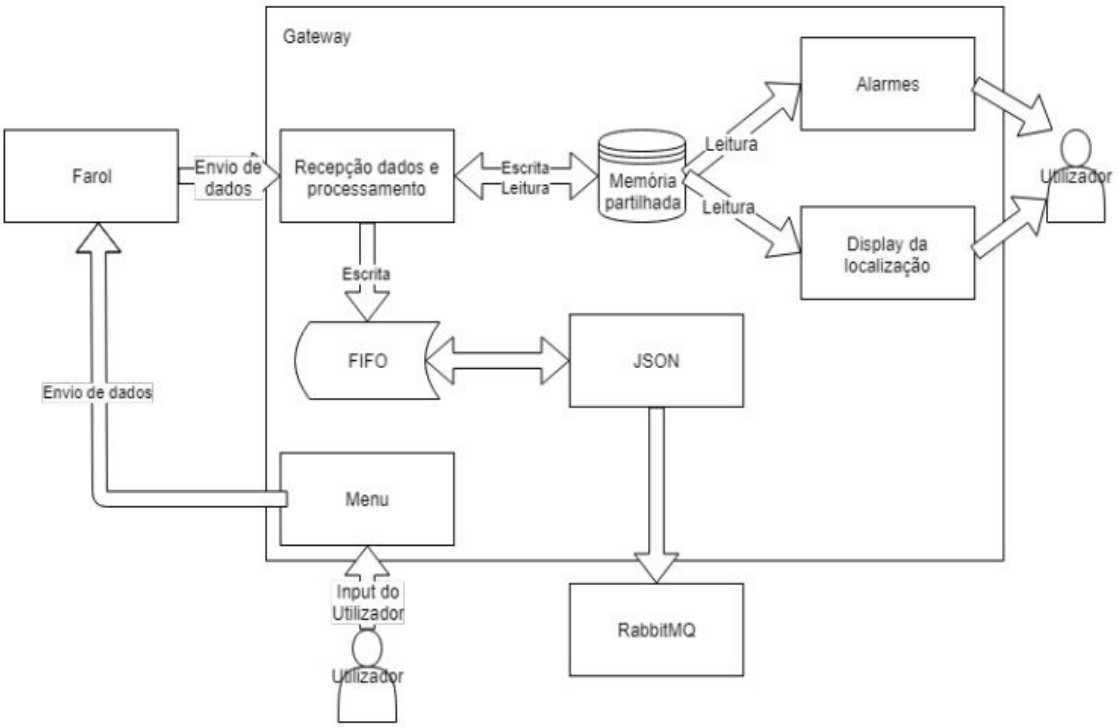

Figura 2.5: Diagrama de blocos da gateway [5]

A figura 2.5 representa o fluxograma da gateway. O algoritmo da gateway é composto por cinco threads:

- **Receção de dados e processamento** - Esta thread é responsável por receber os dados produzidos pela rede e, posteriormente, pelo seu processamento. Quando os dados são recebidos é obtido um timestamp e quando é feito o processamento dos dados são executados os algoritmos de localização. Após isto, os dados processados são guardados, na memória partilhada e na fila First In, First Out (FIFO), com o timestamp correspondente.
- **Alarmes** - Esta thread é responsável por executar algoritmos de averiguação dos dados de modo a ser capaz de identificar casos de anomalias. Caso seja encontrada uma situação anormal esta thread gera um alarme.
- **Display da localização** - Esta thread tem como finalidade mostrar visualmente a localização das coleiras e dos faróis, em tempo real, sem que seja necessário recorrer à interface web ou mesmo quando a conexão a Internet não é possível. Além disso, é também uma boa ferramenta para fazer o *debugging* dos algoritmos de localização.

- **Envio de dados processados** - Esta thread tem o propósito de enviar os dados contidos na fila FIFO para um servidor usando mensagem do tipo JavaScript Object Notation (JSON). Este envio de dados apenas é realizado quando existe conexão a Internet através do protocolo Advanced Message Queuing Protocol (AMQP). Caso aconteça a transferência dos dados, estes são retirados da FIFO. Para o caso o servidor usado é o RabbitMQ [7].
- **Menu** - Esta thread permite ao utilizador o envio de comandos para as coleiras, de modo a que estas executem operações especificas como ativar/desativar algoritmos ou condições de atuação.

Além disso a gateway contém ainda um espaço de *buffering* que é partilhada pelas três primeiras *threads* descritas acima e uma fila FIFO que guarda os dados processados enquanto estes não são enviados para o servidor.

2.2.4 Comunicações

Como já foi referido anteriormente, os faróis são responsáveis por sincronizar as comunicações e retransmitir os dados gerados pelas coleiras. Desta forma, existem três tipo de mensagens a serem trocadas entre os nós da rede.

- **Beacon Synchro (BS)** - Este tipo de mensagem é usado para registar/emparelhar novos dispositivos.
- **Collar to Beacon (C2B)** - Este tipo de mensagem é usado para transmitir os dados gerados das coleiras para os faróis.
- **Beacon to Beacon (B2B)** - Este tipo de mensagem é usado para transmitir os dados de farol em farol.

Uma vez que pode ser necessária a adição de novos dispositivos após a instalação da rede, é necessário que estes consigam pedir a sua admissão na rede. Além disso, a transmissão de mensagens das coleiras e dos faróis deve ser feita em momentos diferentes, de modo a que se tente evitar colisões dos pacotes enviados. Assim o sistema de comunicações proposto pelo projeto SheepIT é baseado em Time Division Multiple Access (TDMA) de forma a minimizar também o consumo de energia. A sua arquitetura e implementação são descritas em mais detalhe em [2] e [8].

O sistema proposto é um sistema rotativo de Micro-ciclo (μC) onde cada um destes μC está associado a um dos tipos de mensagens apresentadas acima. À sequência de μC que é repetida ao longo do tempo dá-se o nome de Macro-ciclo (MC) (figura 2.6).

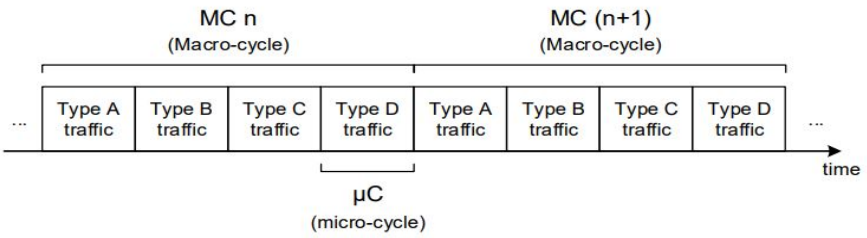

Figura 2.6: Exemplo ilustrativo do Macro-ciclo (MC) e do Micro-ciclo (μC)

Por sua vez, cada micro-ciclo é composto por três janelas temporais, como é possível observar na figura 2.7. Todos os micro-ciclos começam com uma janela de sincronização (SW) onde todos os dispositivos da rede devem estar ativos. Neste ponto, as coleiras sincronizam-se com os faróis e estes sincronizam-se entre si e, para além disso, é também anunciado o tipo de dados que vão ser transmitidos.

Após a sincronização estar concluída, começa a janela dedicada ao processamento de dados (TAW) onde não existem comunicações de rádio. A duração desta janela é igual para todos os dispositivos sendo que são executadas diferentes operações nos faróis e nas coleiras. As operações efetuadas pelos faróis nesta janela consistem essencialmente em identificar o tipo do próximo micro-ciclo, atualizar as tabelas internas e criar novos pacotes. Quanto às coleiras, estas lêem os valores fornecidos pelos sensores e executam o algoritmo de controlo de postura.

Por fim, o micro-ciclo é concluído com uma janela dedicada à transmissão dos dados correspondentes ao tipo de tráfego previamente anunciados.

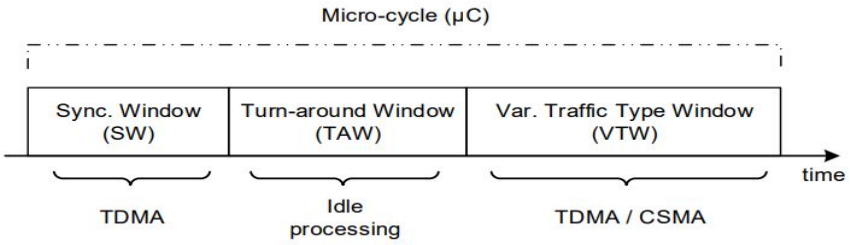

Figura 2.7: Estrutura do Micro-ciclo (μC)

É importante também referir que o valor de RSSI usado para a localização é obtido pelas coleiras aquando a receção dos pacotes enviados pelos faróis, na fase de sincronização (SW).

2.3 Sistema de localização SheepIT

O trabalho desta dissertação teve por base o protótipo de um sistema de localização desenvolvido por José Pereira, na sua dissertação de mestrado [5]. Este sistema de localização é composto por dois mecanismos de localização diferentes para os dois tipos de nós da rede. A figura 2.8 representa o sistema de localização proposto.

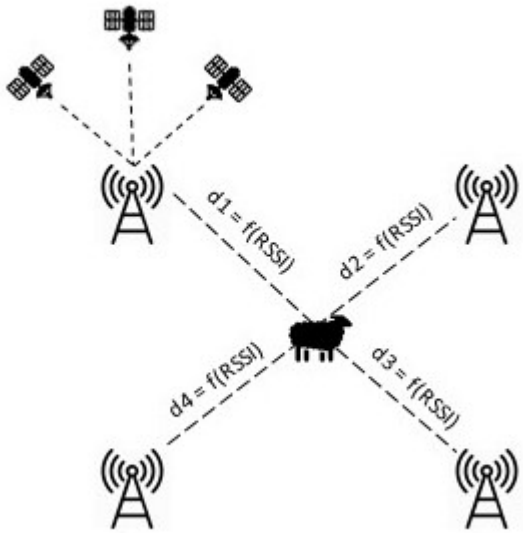

Figura 2.8: Ilustração do sistema de localização SheepIT

O mecanismo de localização dos faróis passa pela obtenção da localização absoluta utilizando um módulo GPS. O módulo GPS escolhido foi o BN-880 [6], sendo que esta escolha foi feita de acordo com um teste onde vários módulos, em funcionamento contínuo, foram colocados em céu aberto a obter a sua posição de segundo a segundo de modo a se conseguir estudar a dispersão dos vários valores de posição.

Uma vez que um dos requisitos do projeto SheepIT é a autonomia dos dispositivos, o funcionamento contínuo do GPS nos faróis não é viável. De modo a reduzir-se o consumo de energia sem prejudicar a precisão da localização foi realizado outro teste, onde foi variado o tempo em que o módulo estava ligado e desligado, e foi calculado o erro e o consumo energético da localização. Este teste teve como propósito definir o método de aquisição dos valores da localização do GPS. Assim, os melhores valores de precisão obtidos, tendo em conta o consumo de energia, foram quando o GPS estava desligado por 120 minutos e ligado durante 1 minuto, sendo que apresentava um consumo de energia de 0.41mAh e um erro médio de 3.92m.

O mecanismo de localização das coleiras permite calcular a posição absoluta destas sem que seja usado um módulo GPS. Este mecanismo é constituído por algoritmos de trilateração, dos quais se obtém uma localização relativa das coleiras baseada em nos valores de RSSI obtidos na comunicação entre os dispositivos. Esta localização relativa, em conjunto com a localização absoluta dos faróis, permite o cálculo da localização absoluta das coleiras.

Foram implementados dois algoritmos de trilateração para o cálculo da localização das coleiras, o algoritmo da distância mais próxima e o algoritmo da interseção de linhas.

O algoritmo da distância mais próxima está representado nas figuras 2.9 e 3.6. Este algoritmo começa por verificar se existem pelo menos 3 valores de RSSI de faróis distintos disponíveis para a trilateração. Caso isto se confirme, são calculadas as distâncias correspondentes aos valores de RSSI recebidos e é criada uma estrutura onde são guardados os dados das circunferências usadas na trilateração, sendo que as coordenadas dos seus centros são as coordenadas dos faróis dos quais foram recebidos os valores de RSSI e os seus raios são as

distâncias calculadas.

Posto isto, são calculadas as interseções de todas as combinações das circunferências e depois é verificado se existem pelo menos 6 interseções. Se isso se confirmar é calculada, para cada ponto, a distância mínima entre esse mesmo ponto e os restantes pontos da interseção das circunferências e de seguida são escolhidos os pontos com as distâncias mais pequenas para se estimar a localização da coleira, ou seja, os pontos mais próximos. O número de pontos escolhidos depende do número de faróis usados para o cálculo da localização.

Finalmente, com os pontos obtidos é realizada uma média de modo a estimar-se uma localização.

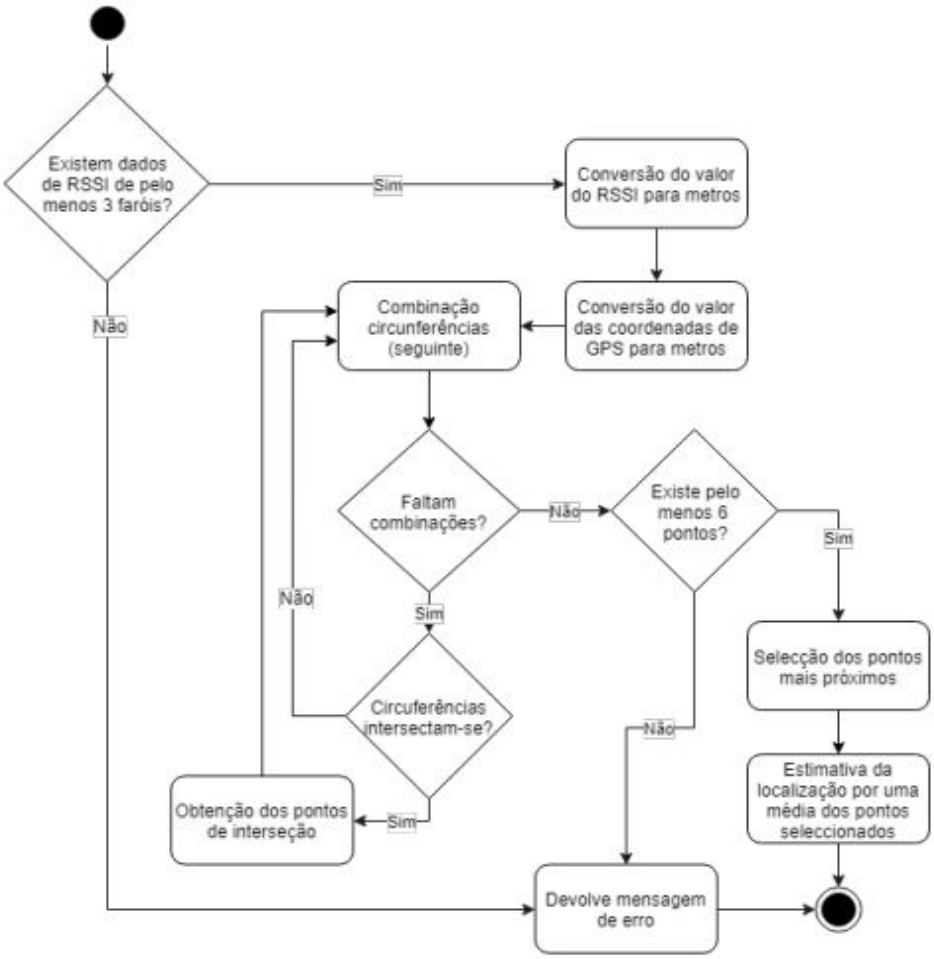

Figura 2.9: Fluxograma do algoritmo da distância mais próxima [5]

O algoritmo da interseção de linhas está representado nas figuras 2.10 e 3.7. Como é possível observar a partir dos fluxogramas dos algoritmos de trilateração, tanto o algoritmo da distância mais próxima como este algoritmo começam por verificar se existem pelo menos 3 valores de RSSI disponíveis e calcular os pontos de interseção das circunferências geradas.

Para cada par de pontos gerado pelas interseções das circunferências é gerada uma linha que une esses pontos e é verificado se existem pelo menos duas linhas. Em caso de isto se confirmar, são calculados os pontos de interseção de todas as combinações de linhas,os quais são

guardados numa estrutura. Depois de todas as combinações serem feitas, são removidos dos pontos calculados aqueles cuja localização não esteja contida em nenhuma das circunferências geradas.

Por fim, é verificado se existe pelo menos um ponto válido e, caso isso se confirme, é estimada uma localização da coleira fazendo uma média dos pontos restantes.

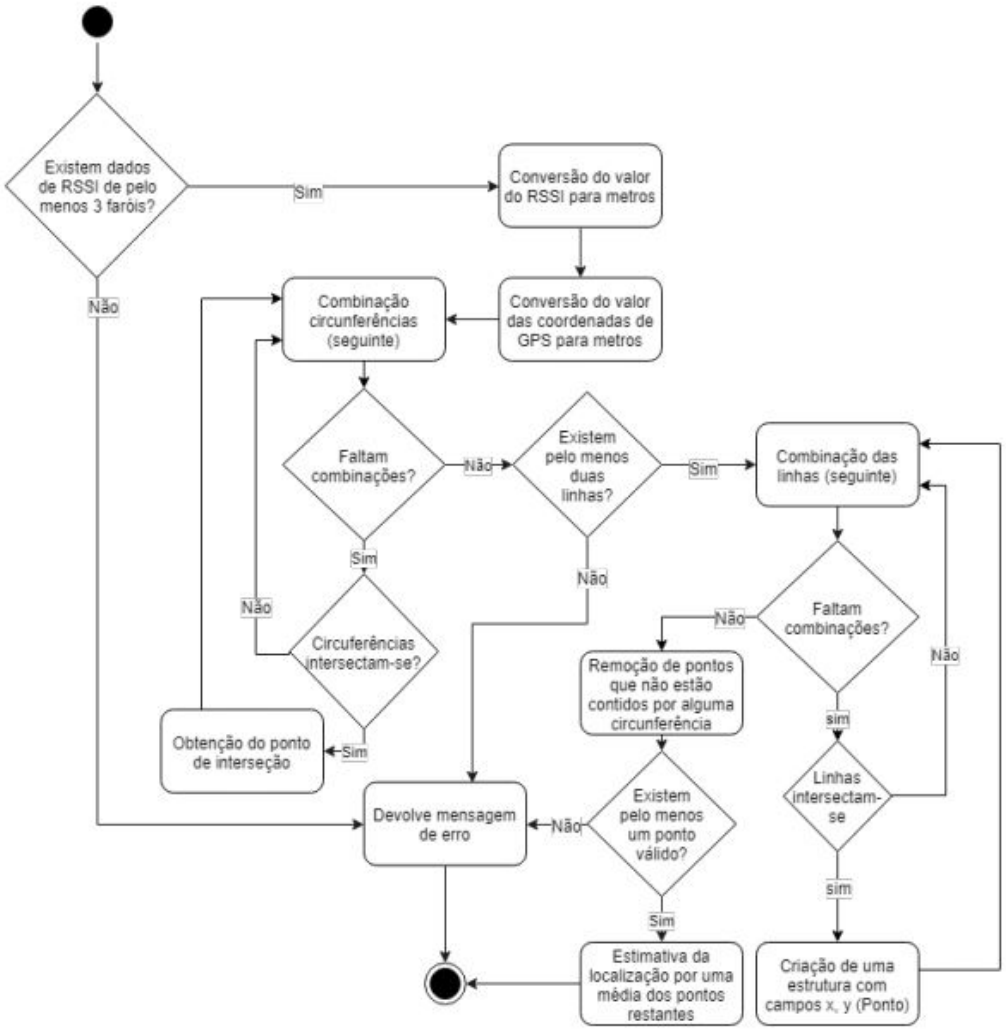

Figura 2.10: Fluxograma do algoritmo da interseção de linhas [5]

Foi verificado que não é possível estimar a localização de uma coleira com nenhum destes métodos quando as circunferências geradas pelas distâncias calculadas entre a coleira e os faróis não se intersetam. De forma a minimizar esta limitação foi desenvolvido um algoritmo avançado de localização.

O algoritmo avançado de localização está representado na figura 2.11. Este algoritmo começa por executar os dois algoritmos de trilateração previamente apresentados. Após a execução destes é verificado se as localizações estimadas por estes são ambas válidas. No caso de serem ambas válidas é estimada uma localização para a coleira a partir da média das localizações estimadas.

Por outro lado, no caso desta primeira condição não se confirmar, é verificado se existe alguma estimativa dos algoritmos de trilateração que seja valida. Caso esta condição se confirme, então a estimativa da posição da coleira é dada pela estimativa válida de um dos algoritmos.

Se porventura nenhuma das condições anteriores se confirmar, a estimativa da localização da coleira é feita através da análise direta dos valores de RSSI recebidos pela rede. Para tal, é verificado se o maior valor de RSSI recebido está acima de um limitar *thr1*. Este limiar *thr1* define a zona a partir do qual a sensibilidade do modelo de propagação usado para estimar distâncias é reduzida. Assim, se esta condição se verificar é considerado que a coleira está muito próxima do farol que devolveu esse valor de RSSI sendo a posição da coleira dada pela posição desse farol. Finalmente, caso o melhor valor de RSSI não seja maior que o limiar *thr1* então é calculado um vetor unitário com origem em P_B e direção P_B para P_A, onde o ponto P_A é dado pela posição do farol com melhor RSSI e P_B é dado pela equação seguinte:

$$P_B = A - B \tag{2.1}$$

onde, A é a soma de todas as posições dos faróis com valor de RSSI abaixo de um limiar *thr2* e B é a subtração de todas as posições dos faróis com valor de RSSI igual ou acima do limiar *thr2*. O limiar *thr2* define a zona a partir do qual a sensibilidade do modelo de propagação é grande e representa também o limiar de proximidade do limite de cobertura do rádio. Assim sendo a posição da coleira é estimada a partir da equação seguinte:

$$P_e = P_A + D \times V \tag{2.2}$$

onde, D é a distância associada ao melhor valor de RSSI recebido.

Os valores de *thr1* e *thr2* obtidos após a análise do modelo de RSSI vs Distância foram de *thr1* = -42 dBm e *thr2* = -63 dBm que representam uma distância de 5 e 30 m respetivamente.

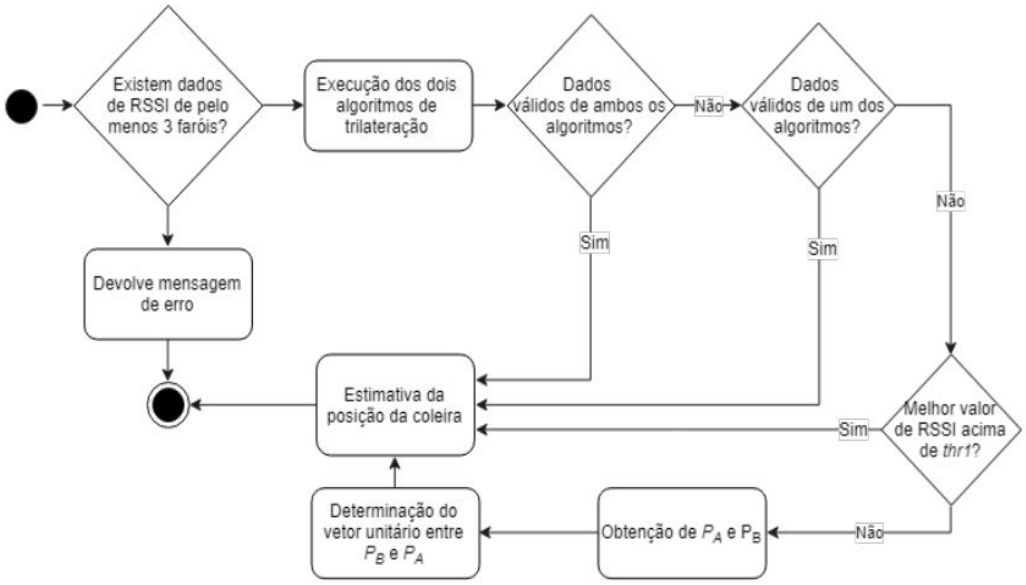

Figura 2.11: Fluxograma do algoritmo avançado de localização [5]

Posteriormente o autor destes algoritmos desenvolveu uma variação do algoritmo avançado de localização onde é usada informação a cerca da movimentação do animal.

Esta versão do algoritmo começa por executar um dos algoritmos de trilateração apresentados anteriormente obtendo uma estimativa preliminar da posição da coleira. De seguida é verificado qual o estado do animal que foi registado, isto é, é verificado em qual dos estados *STOPPED*, *ACTIVE* ou *RUNNING* estava o animal no momento da leitura dos sensores. Assim, consoante o estado que foi verificado são executadas diferentes ações:

- ***STOPPED*** - Este estado representa o estado de quando o animal está parado e por isso neste estado é feita uma média das 10 últimas posições registadas, incluindo a nova estimativa da posição.
- ***ACTIVE*** - Neste estado é feita uma média das 3 últimas posições registada incluindo a nova estimativa da posição.
- ***RUNNING*** - Neste estado é verificado se a distância entre a posição atual estimada pelo algoritmo de trilateração e a posição anterior registada é maior do que um valor de distância estabelecido correspondente à máxima distância que o animal pode percorrer entre cada nova estimativa da posição. Para o efeito o valor estabelecido é de 10m. Caso esta condição se verifique é calculada uma posição com recurso a um vetor semelhante ao calculado na versão anterior do algoritmo avançado de localização, mas com uma direção e distância diferentes. A direção do vetor calculado é de P_A para P_B, onde P_A é a última posição registada e P_B é a estimativa da posição atual, e a distância usada é de 10m.

Este sistema de localização obteve erros de localização entre os 7m e os 20m quando testado para uma topologia de 4 faróis em forma de quadrado e espaçados de 30m . Uma vez que estes valores de erro foram obtidos num ambiente aberto livre de obstáculos, é possível afirmar que estes valores e erro serão maiores quando este sistema for implementado num ambiente real (vinha) devido à quantidade de obstáculos que existirão entre as coleiras e os faróis. No Capítulo 4 encontra-se descrita a arquitetura do novo sistema de localização, no qual foram desenvolvidos mecanismos para minimizar o erro de localização através da redução do erro das distancias estimadas.

Capítulo 3

Sistemas de rastreio e localização animal

Neste capítulo começam por ser apresentados alguns sistemas focados na localização animal. Depois são também apresentados alguns sistemas de localização para redes de sensores sem fios, ou em inglês Wireless Sensor Network (WSN), que usam medidas de RSSI. Por fim são abordadas algumas tecnologias e métodos que são usualmente utilizados para determinar a localização de um dispositivo.

3.1 Sistemas de localização animal

A localização animal tem sido uma área de investigação com grande desenvolvimento principalmente no que toca a localização de gado. Com a redução do número de pastores foi necessário encontrar soluções de forma a poder monitorizar-se as áreas de pasto. Esta necessidade levou ao desenvolvimento de varias soluções, sendo a maioria delas baseada no Sistema de Navegação Global por Satélite (Global Navigation Satellite System (GNSS)), mais especificamente na tecnologia Global Positioning System (GPS).

Tanto a nível comercial como na literatura podem ser encontrados sistemas que permitem localização animal, como por exemplo o e-Pasto [14] [15], o OzTrack [16], digitanimal [18], eShepherd [20] e o Nofence [21].

O sistema e-Pasto permite localizar gado de pastagem em montanhas e está dividido em duas partes: dispositivo de localização, e uma plataforma de interface de utilizador que permite visualizar as localizações do animal num intervalo de tempo e permite ainda definir cercas virtuais de modo a delimitar a área permitida para pastagem. A localização é obtida uma vez por hora a partir de um módulo GPS conseguindo assim uma autonomia de 7 meses. Em [14] é apresentada uma solução para diminuir o consumo energético. Esta solução passa por incorporar um acelerômetro no dispositivo de localização de modo a que a medição da localização só seja feita quando um movimento significativo do animal é detetado por este. A figura 3.1 ilustra o dispositivo de localização e a interface de utilizador.

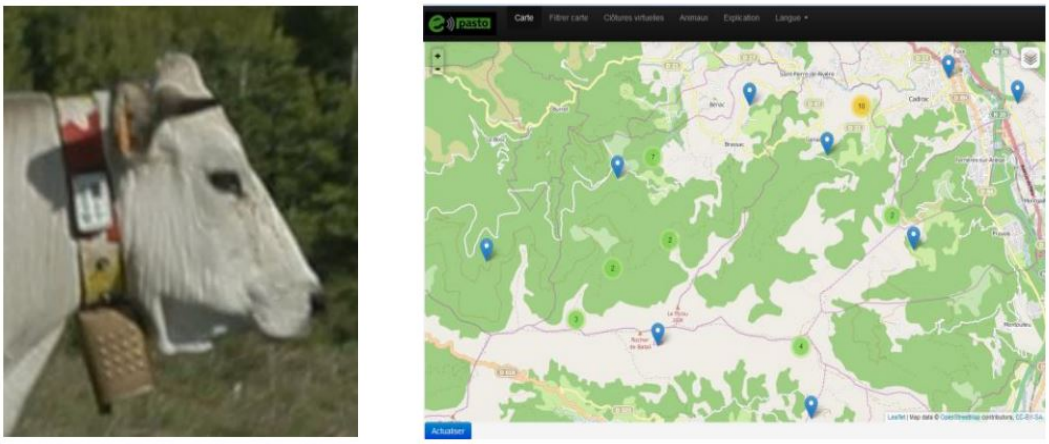

Figura 3.1: Dispositivo de localização e a interface de utilizador [15]

O sistema OzTrack é uma plataforma que permite guardar, processar e analisar dados de localização produzidos por dispositivos de telemetria usados por animais. Esta plataforma foi utilizada por Hunter et al. [17] num caso de estudo onde se pretendia estudar o comportamento de crocodilos. A plataforma permite ao utilizador criar diversos projetos e adicionar dados de um ou mais animais ao projeto em ficheiros do tipo CSV ou Excel. Posteriormente pode-se selecionar um projeto e os animais que se pretendem analisar. Os dados da localização dos animais são presentados com diferentes cores para cada um em cima de uma imagem de satélite do GoogleMap, como é possível ver na figura 3.2. Para além da visualização dos dados da localização a plataforma também permite que seja visualizado dados estatísticos calculados por esta.

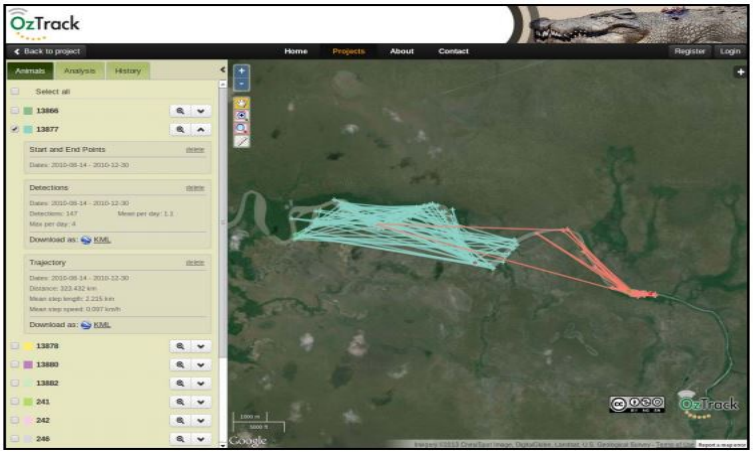

Figura 3.2: Plataforma OzTrack

O sistema digitanimal permite localizar diferentes espécies de animais tais como gado bovino, ovino, caprino, equino, linces, pássaros, entre outros. A localização é feita a partir de um módulo GPS contido nas coleiras usadas pelos animais (figura 3.3). A comunicação das mensagens das coleiras é feita com recurso a duas tecnologias diferentes, nomeadamente, conexão de rede Sigfox e conexão de rede GSM. O sistema compreende ainda um aplicativo

que permite visualizar a localização dos animais, podendo este ser acedido através de qualquer navegador web, e também aplicativos para Android e iOS. Em [19] é apresentado um sistema de localização de mais baixo custo que faz uso das coleiras do sistema digitanimal em conjunto com *tags* de Bluetooth de baixo custo. A localização da solução proposta é feita a cada cada 30 minutos, onde o módulo de GPS das coleiras sai do modo inativo e obtém novas coordenadas. Ao mesmo tempo o módulo de comunicação BLE é ativado e durante alguns segundos verifica se é detetada alguma mensagem enviada pelas *tags* de Bluetooth e guarda-as. Após isto a coleira envia os dados do GPS e do módulo BLE, através do módulo de comunicações Sigfox, para um servidor onde posteriormente serão analisados.

Figura 3.3: Coleira do digitanimal

O sistema eShepherd [20] para além de permitir a localização do animal permite que sejam criadas cercas virtuais de modo a limitar a zona de pastagem. Este sistema é usado em vacas e é composto por uma coleira com um módulo GPS e por uma aplicação móvel que permite criar cercas virtuais, mover e monitorizar o gado. Os limites da cerca podem ser definidos usando o Google Maps num computador ou tablet e depois são enviados para a coleira através de tecnologia rádio.

O sistema Nofence [21], tal como o anterior, permite localizar os animais e limitar a sua área de pastagem criando cerca virtuais. Este sistema pode ser usado em cabras, vacas e ovelhas. A localização é feita através de um módulo GPS contido nas coleiras e os limites de pastagem podem ser definidos usando o mapa da aplicação do sistema. O número de medições da posição depende da distância a que o animal se encontra do limite da cerca, isto é, quando este se encontra perto dos limites o número de medições do módulo GPS é maior do que no resto do tempo, permitindo assim um aumento na autonomia da bateria. As coleiras estão ainda equipadas com um módulo Bluetooth que em conjunto com os faróis do sistema, equipados com um módulo de Bluetooth, são usados para identificar desligar o módulo GPS da coleira quando esta está dento de um edifício.

3.2 Sistemas de rastreio

Uma vez que esta dissertação propõe melhorar o sistema de localização presente no projeto SheepIT, um dos requisitos é que o consumo de energia seja o menor possível de modo a obter uma maior autonomia do sistema e que o custo da tecnologia usada seja também compatível.

Algumas das soluções apresentadas na secção anterior apresentam uma grande autonomia utilizando GPS. Contudo, a autonomia dessas soluções é conseguida devido a duas particularidades que vão contra os requisitos do projeto SheepIT. Uma primeira é relativa ao tamanho da coleira que como são soluções que são usadas em animais de grande porte permitem que a coleira seja maior e por sua vez que as baterias usadas sejam também maiores que no projeto SheepIT. A segunda é relativa ao intervalo de tempo entre cada obtenção da localização. Enquanto que as soluções apresentadas acima obtêm a localização com grandes intervalos de tempo, o sistema de localização do SheepIT é um sistema de localização em tempo real, ou seja, era necessário que fossem feitas mais medições da localização por unidade de tempo levando assim a que o consumo de energia fosse maior.

Por estas razões um sistema que obtenha a localização a partir de um módulo GPS não é viável e, portanto, são apresentados alguns sistemas que fazem uso da tecnologia RSSI para estimar a posição de um nó numa rede de sensores sem fios.

Em [10] é apresentado um sistema que é utilizado para monitorizar um oleoduto com o comprimento de 1km. Para isso foram fixados ao longo do oleoduto dois tipos de nós capazes de transmitir dados de temperatura, luz e humidade: nós ancora ou farol cuja localização é conhecida, e nós cegos, cuja sua localização é desconhecida. O sistema pretende então localizar a posição dos nós cegos no oleoduto utilizando valores de RSSI em conjunto com um método de trilateração. Foram usados quatro *crossbow's TelosB node TPR2420* para a realização dos testes, onde foram medidos erros de 0.74m e 0.56m quando foram usados três e quatro nós ancora respetivamente.

Um sistema de localização de um robô móvel em ambiente externo é apresentado por Graefenstein e Bouzouraa *et al.* [11]. Este artigo apresenta dois algoritmos de localização baseados em RSSI para redes sem fios de baixa potencia com a norma IEEE 802.15.4. O primeiro algoritmo utilizado na localização do robô é uma algoritmo probabilístico e tem o nome probability torus localization (PTL) devido à interpretação geométrica da densidade de probabilidade resultar de um toro em cada nó. O segundo algoritmo tem o nome de Sequence-Based localization (SBL) e localiza um nó a partir de uma sequência ascendente de valores de RSSI, isto é, a partir dos valores de RSSI é possível obter distâncias entre nós de referência e assim definir áreas onde essa distância diminui. Foram efetuados testes para ambos os algoritmos sendo obtido erros de localização de 0.94m e 2.17m para os algoritmos PTL e SBL, respetivamente. Com recurso a *sensor fusion* de dados de odometria, de ultra-sons e de um mapa do local, os autores obtiveram um aumento em 67% na precisão obtendo um erro de 0.32m na localização.

Em [12] são propostos dois algoritmos de localização para uma rede de sensores Zigbee em ambiente externo. Ambos os algoritmos são baseados em estimativas de distância a partir de valores de RSSI e trilateração. Para estimar os valores de distância foi considerado um modelo de propagação que em espaço aberto é descrito por $RSSI = -10\eta log_{10}(d) + A$. Os algoritmos propostos são o Circle Intersection Algorithm (CIA) e o Line Intersection Algorithm (LIA). Ambos começam por desenhar círculos com raio igual às distâncias estimadas. O primeiro calcula a posição fazendo uma média dos pontos de interseção dos círculos enquanto que o segundo desenha linhas de interseção entre cada dois círculos e depois calcula a posição fazendo

a média dos pontos de interseção das linhas. Finalmente foram realizados testes para ambos os algoritmos onde foram colocados 4 nós, cuja sua localização é conhecida, em forma de um quadrado e um outro nó, cuja sua localização se prende determinar, em 20 sítios aleatórios do quadrado. Este procedimento foi feito para diferentes distâncias entre os nós de referência e foi obtido um erro de localização médio menor que 3m.

Em [13] é apresentado um outro sistema de localização para redes sem fios em ambiente externo baseado em medições de RSSI e trilateração. O módulo de comunicação usado para obter os valores de RSSI foi o CC2530 RF e, portanto, foi estudada experimentalmente a variação dos valores de RSSI com a distância. A partir da linearização e interpolação da medidas foi obtida uma equação capaz de converter os valores de RSSI em distância, $d = 10^{(\frac{-51.1516-RSSI}{25.9879})}$. Após ser achada esta equação foram realizados testes em ambiente real onde foi aplicada a trilateração aos valores de distância obtidos por esta equação e foi obtido um erro médio de 4.9m.

3.3 Localização

Nesta secção são apresentadas diversas tecnologias e métodos de localização. Esta divisão é feita de modo a separar dois processos essenciais à estimação de uma localização. Enquanto que a tecnologias apresentadas permitem o cálculo da distância (ou ângulo) entre os dispositivos, estes métodos permitem o cálculo de localização absoluta fazendo uso das distâncias calculadas no processo anterior. Em [22] são apresentadas algumas tecnologias e métodos de localização, as quais se descrevem resumidamente abaixo.

3.3.1 Tecnologias

Para se poder estimar a localização de um nó é necessário começar por se calcular a distância (ou ângulo) entre este nó e o nó cuja sua localização é conhecida. Existem, portanto, diversas tecnologias capazes de estimar uma distância, das quais se destacam tecnologias baseadas em medições de ângulos, tempo de propagação e em propriedades do sinal.

AoA

A tecnologia Angle of Arrival (AoA) [23], ou ângulo de chegada, determina o ângulo de chegada do sinal proveniente de um dispositivo cuja sua localização é desconhecida. O valor de ângulo é medido através da diferença de fase do sinal recebido em diferentes antenas. Para estimar a posição em 2-D esta tecnologia tem a vantagem de apenas precisar de dois faróis. Contudo, esta tecnologia tem a desvantagem de a precisão da localização diminuir quando existem reflexões de sinal e quando o nó móvel se afasta dos nós fixos, e de necessitar de hardware complexo, como antenas altamente direcionais.

ToA

O mecanismo Time of Arrival (ToA) [24], ou tempo de chegada, determina o tempo de propagação do sinal desde o emissor até ao recetor. A distância entre o nó emissor e o nó recetor é diretamente proporcional ao tempo de propagação, sendo assim facilmente calculada através da multiplicação da velocidade de propagação do sinal com o tempo obtido. No entanto, para calcular o tempo de propagação é essencial saber em que instante foi enviado o sinal,

logo é preciso que ambos os relógios do emissor e do recetor sejam extremamente precisos e sincronizados.

RToF
A tecnologia Round-Trip Time of Flight (RToF) [25], ou tempo de ida e volta, determina o tempo de propagação do sinal desde o emissor até ao recetor e do recetor de volta para o emissor. O cálculo da distância é feito de forma semelhante ao do mecanismo ToA com a diferença de que o valor do tempo usado para o cálculo é metade do valor de tempo obtido. Ao contrário do ToA que determina o tempo de voo com recurso a dois relógios de dispositivos diferentes, no RToF o tempo de voo é determinado apenas com um relógio. Por esta razão, esta tecnologia tem a vantagem de não ser necessária uma sincronização dos relógios dos diferentes dispositivos e de não ser necessário comunicar o instante inicial da transmissão do sinal. Contudo esta tecnologia pode apresentar latências indesejadas quando são feitas medições consecutivas de distância para vários dispositivos.

TDoA
A tecnologia Time Difference of Arrival (TDoA) [26], ou diferença de tempos de chegada, estima a distância entre o dispositivo móvel e os faróis através da diferença dos tempos de chegada de um sinal proveniente do dispositivo móvel. Como esta tecnologia usa apenas os instantes de chegada de um sinal, cujo instante inicial de transmissão é desconhecido, não é necessária uma sincronização de todos os relógios dos diferentes dispositivos (moveis e fixos) mas sim dos relógios dos dispositivos recetores. Assim sendo, os emissores ao poderem transmitir um sinal a qualquer instante faz com que os recetores tenham de estar continuamente a espera de receberem um sinal, o que faz com que o consumo de energia por parte destes aumente.

RSS/RSSI
Esta tecnologia permite estimar a distância entre dois nós a partir da atenuação da intensidade do sinal de rádio emitido, isto é, quando um rádio recebe um sinal este consegue medir a sua potência ou intensidade na sua receção e, comparando este valor com o valor da potência transmitida, é possível estimar uma distância. A este valor de potência do sinal recebido dá-se o nome Received Signal Strength (RSS) sendo por norma a sua unidade de medida dBm. Contudo, existem rádios que não fornecem este valor de intensidade de sinal, mas sim um indicador desta, a que se dá o nome de Received Signal Strength Indication (RSSI). Este valor de RSSI de acordo com o padrão IEEE 802.11 pode estar numa escala de 0 até 255 sendo que cada fabricante pode definir o seu próprio valor de RSSI máximo.

Com alguns testes empíricos é possível determinar como é que o RSS varia com a distância. Contudo, os valores de RSS são gravemente afetados pelo ruído, ruído este que pode ser devido a diversas causas tais como perdas de sinal por causa de reflexões ou refrações do sinal. Existem três modelos matemáticos que permitem modelar estas perdas, descritos em [27], dos quais se destaca um que é apresentado de seguida:

- *Log-normal shadow model* - Este modelo é o mais genérico tanto para ambientes interiores com exteriores, permitindo que alguns parâmetros sejam calibrados de acordo com o ambiente do sistema. O modelo é dado pela equação seguinte:

$$P_L(d)[dB] = \overline{P_L}(d) + X_\sigma = \overline{P_L}(d_0) + 10\eta log(\frac{d}{d_0}) + X_\sigma \tag{3.1}$$

onde, $P_L(d_0)$ é um valor de potência medido experimentalmente a uma distância de d_0, η é um valor de perda que depende do ambiente de propagação e X_σ é uma variável aleatória Gaussiana com média nula.

Para além das perdas, diferenças no hardware de rádio para rádio, presença de obstáculos entre os dois rádios e até mesmo variações das condições atmosféricas, são causas de perturbação nos valores de RSS. Em consequência destas perturbações no valor de RSS o valor da distância calculado sofrerá de um erro, que aumenta com a distância.

Por fim, esta tecnologia apesar de apresentar maiores problemas de precisão e exactidão em relação as tecnologias apresentadas anteriormente, tem a vantagem de não precisar de *hardware* adicional.

3.3.2 Métodos de localização

Uma vez calculada a distância (ou ângulo) entre os dispositivos, é necessário combinar os valores calculados de maneira a obter uma localização. Existem vários métodos para inferir a posição de um nó a partir de medições, nomeadamente trilateração, triangulação e multilateração.

Trilateração

O método de trilateração permite o cálculo da posição a partir de estimativas de distância entre uma estação móvel e pelo menos três estações base, cuja sua localização é conhecida. Este método é normalmente usado em conjunto com as tecnologias ToA. RToF e RSSI.

Sabendo a posição das estações base e a distância a que estas se encontram da estação móvel, são geradas três circunferências cujos centros são dados pelas coordenadas das estações base e os raios pelas distâncias estimadas. As equações das circunferências são dadas pela equação 3.2.

$$D_i^2 = (x - x_i)^2 + (y - y_i)^2 \qquad , i = \{1, \quad 2, \quad 3\} \tag{3.2}$$

onde D_i é a distância estimada entre a estação móvel e a estação base i, (x,y) são as coordenadas da estação móvel e (x_i,y_i) são as coordenadas da estação base i.

No caso ideal, a posição seria estimada pelo ponto de interseção das três circunferências como é possível observar na figura 3.4.

Contudo, em ambiente real as distâncias estimadas são afetadas de erro fazendo com que as três circunferências não se intersetem num só ponto, como é possível ver na figura 3.5, não sendo possível calcular a posição pela interseção das circunferências. Em [28] são apresentados dois algoritmos que propõem resolver este problema e são descritos de seguida:

- Algoritmo da distância mais próxima - Este algoritmo determina a posição a partir dos pontos de interseção interiores, isto é, começa por calcular o par de pontos de interseção obtidos pela interseção de duas circunferências. Dos dois pontos calculados é escolhido aquele que apresenta uma menor distância até a estação base que não foi usada para determinar este par de pontos. De seguida este procedimento é repetido para todas as combinações das circunferências e uma vez calculados todos os pontos interiores, a posição é estimada a partir de uma média das coordenadas dos pontos interiores. A figura 3.6 é uma representação deste algoritmo onde os triângulos são os pontos interiores escolhidos para o cálculo da posição.

- Algoritmo de interseção de linhas - Este algoritmo, à semelhança do anterior, começa por calcular os pontos de interseção par cada par de circunferências. De seguida são traçadas linhas que unem o par de pontos de cada par de circunferências. A posição é então estimada pela interseção destas linhas como é possível ver na figura 3.7;

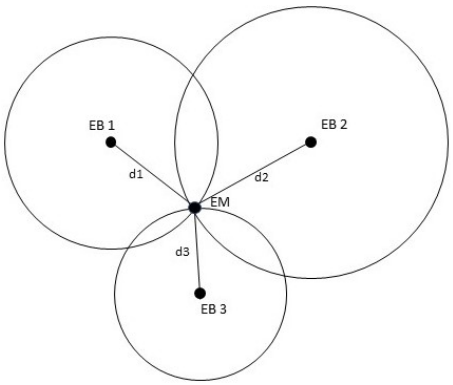

Figura 3.4: Método de trilateração

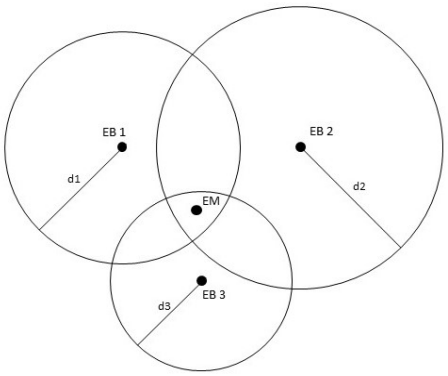

Figura 3.5: Método de trilateração com erro nas distâncias

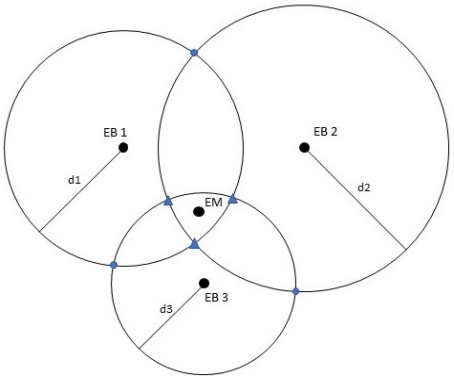

Figura 3.6: Algoritmo da distância mais próxima

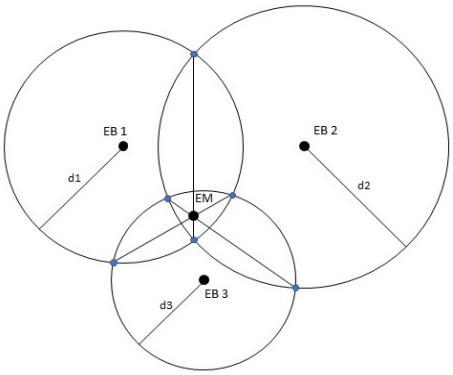

Figura 3.7: Algoritmo da interseção de linhas

Multilaterção

O método de multilateração ou lateração hiperbólica determina a posição de uma estação móvel a partir das diferenças de distância calculadas pela tecnologia TDoA. Para cada par de estações base é medida de diferença de tempo de chegada de um sinal proveniente da estação móvel correspondente a uma diferença de distância. Para cada diferença de distância calculada é possível desenhar uma hipérbole que corresponde às posições possíveis da estação móvel para as quais as diferenças de distância se mantêm, figura 3.8. As equações das hipérboles são dadas por:

$$R_{i,j} = \sqrt{(x_i - x)^2 + (y_i - y)^2} - \sqrt{(x_j - x)^2 + (y_j - y)^2} \tag{3.3}$$

onde $R_{i,j}$ corresponde à diferença de distância calculada (x_i, y_i) e (x_j, y_j) são as coordenadas das estações base usadas para o cálculo da diferença de distância e (x, y) corresponde as coordenadas da estação móvel.

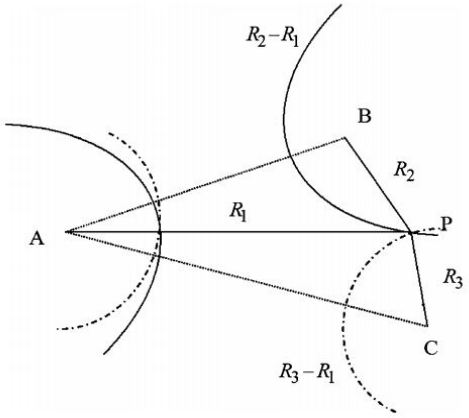

Figura 3.8: Método de multilateração [29]

Por fim, o cálculo da posição da estação móvel, ponto P, é feito a partir da interseção de duas ou mais hipérboles como se observa na figura 3.8.

Triangulação
O método de triangulação determina a posição de uma estação móvel a partir da direção de um sinal. Isto é, com recurso à tecnologia AoA mede-se o ângulo de chegada de um sinal.

O cálculo da posição da estação móvel é possível através de duas ou mais medições de ângulos de chegada do sinal como é possível observar na figura 3.9. Com apenas estas duas medições de ângulos, e sabendo a posição das estações base, é possível traçar duas linhas, cujas suas direções são dadas pelos ângulos obtidos, que se intersetam num ponto que corresponde à posição que se pretende estimar.

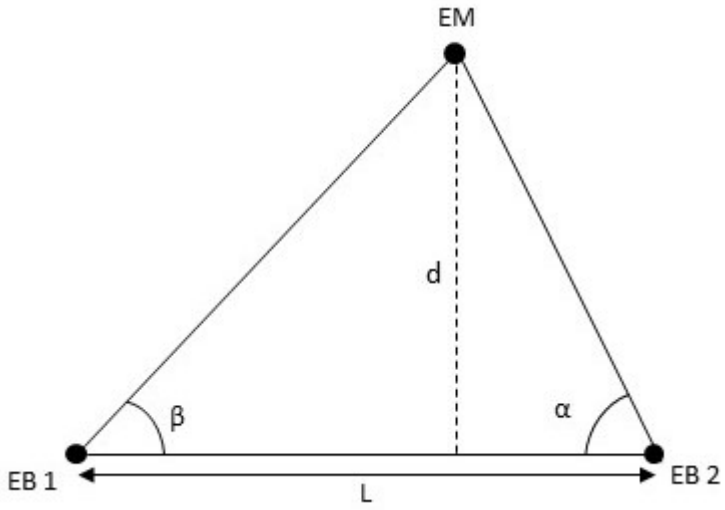

Figura 3.9: Método de triangulação

3.3.3 Filtro de Kalman

Quando se pretende atenuar o efeito do ruído e de outras incertezas é usual recorrer ao uso da ferramenta matemática denominada de Filtro de Kalman (FK) que foi criada por Rudolf Kalman [32].

O filtro de Kalman [33] tem como propósito estimar valores que tendam para os valores reais das grandezas medidas utilizando medições dessas mesmas grandezas, afetadas de ruído e outras incertezas, ao longo do tempo. O filtro de Kalman é um algoritmo recursivo que estima o estado interno de um sistema dinâmico linear a partir de uma serie de medições ruidosas.

O filtro de Kalman apresenta uma vasta diversidade de aplicações, sendo a sua aplicação muito comum a tecnologias militares e espaciais, navegação e rastreamento de objetos. Para além disso, pode ser aplicado também nas áreas da economia, da medicina e da visão por computador. A fim de melhorar a estimação do estado de um sistema recorre-se muitas vezes a um processo de combinação de dados de diferentes sensores, a que se dá o nome de fusão sensorial, sendo o filtro de Kalman uma eficiente forma de fazer esta combinação de dados.

Como já foi referido anteriormente, o filtro de Kalman tem como objetivo produzir estimativas do estado interno de um sistema dinâmico linear e por isso o algoritmo está dividido em duas fases, uma primeira fase de predição, seguindo-se uma fase de correção/atualização. Assim sendo o algoritmo começa pela fase de predição, onde estima as variáveis de estado do sistema assim como as suas incertezas, usando as medidas presente e passadas. Em seguida o algoritmo entra na fase de correção/atualização. Então, quando se obtém um novo valor

da grandeza medida estas estimativas são atualizadas com uma média ponderada dando um maior peso às estimativas mais certas. A estimativa da incerteza é também atualizada de acordo com o peso anteriormente dado. Uma vez que o algoritmo é recursivo, o filtro pode ser usado em tempo real usando apenas as medições presentes e os valores das variáveis de estado anteriores sem que mais nenhuma informação do passado seja necessária, isto é, não é necessário guardar as variáveis de estado para além das variáveis da iteração anterior. A figura 3.10 representa o diagrama de estados do algoritmo explicado anteriormente.

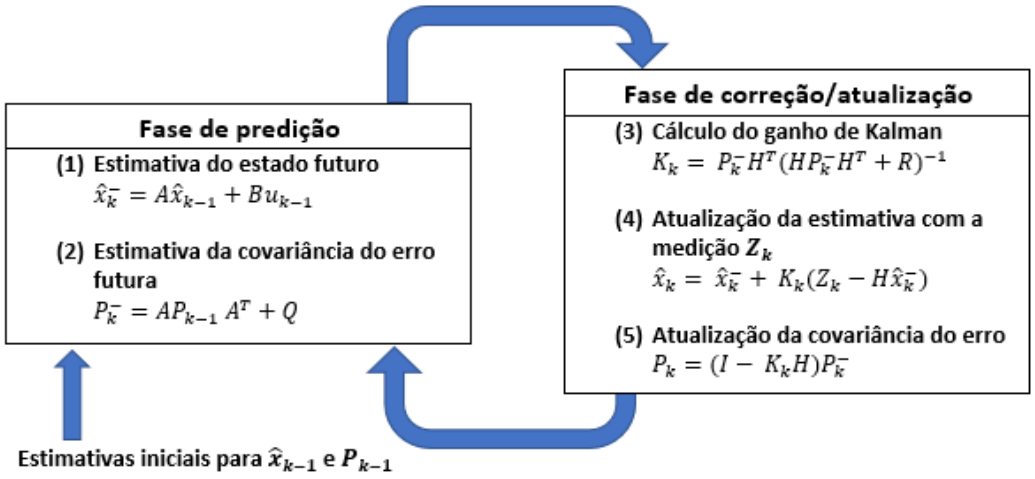

Figura 3.10: Diagrama do Filtro de Kalman

$\widehat{x}_k$ e $\widehat{x}_k^-$ correspondem às estimativas das variáveis de estado atual e futura respetivamente, P_k e P_k^- são as estimativas das incertezas atual e futura respetivamente, Q e R são as matrizes de covariância do ruído de processo e de medição, respetivamente, K_k corresponde ao chamado ganho de Kalman e I corresponde a uma matriz identidade.

A partir das equações (3) e (4) da figura 3.10 é possível observar que à medida que P_k tende para zero o ganho diminui, levando a que a estimativa atualizada do estado do sistema seja dada pela equação (1), ou seja,

$$\lim_{P_k \to 0} = 0 \tag{3.4}$$

e, portanto, a equação (4) passa a ser,

$$\widehat{x}_k = \widehat{x}_k^- \tag{3.5}$$

Por outro lado, à medida que R tende para zero o ganho aumenta. Isto quer dizer que a confiança dada ao valor medido aumenta sendo que, em última instância, a estimativa atualizada do estado do sistema é dada pelo valor medido, ou seja,

$$\lim_{R \to 0} K_k = H^{-1} \tag{3.6}$$

e, portanto, a equação (4) passa a ser,

$$\widehat{x}_k = \widehat{x}_k^- + H^{-1}y_k - H^{-1}H\widehat{x}_k^- \tag{3.7}$$

$$<=> \widehat{x}_k = H^{-1} y_k \tag{3.8}$$

considerando C = 1,

$$x_k = y_k \tag{3.9}$$

3.4 Discussão

Como pode ser observado na Secção 3.1, existem diversas soluções, tanto a nível comercial como na literatura, focadas para a localização animal. Contudo, estas soluções são baseadas em GPS sendo portanto inviáveis para serem implementadas no projeto SheepIT devido ao custo e ao consumo de energia desta tecnologia irem contra os requisitos do sistema de localização. De seguida, na secção 3.2 são apresentados e analisados algumas soluções de localização que recorrem a medidas de RSSI para localizar um nó numa rede de sensores sem fios. Por fim, são apresentadas a tecnologias e métodos mais comuns para efetuar localização.

Capítulo 4

Arquitetura e Implementação

Neste capítulo é descrita a arquitetura e implementação do sistema de localização proposto para o sistema SheepIT. Tendo em conta as limitações do rádio usado nas comunicações é proposto um mecanismo de calibração para o modelo de propagação de sinal usado para estimar distâncias. De seguida são apresentados três mecanismos de filtragem para minimizar o erro das distâncias estimadas.

4.1 Arquitetura da solução proposta

O objetivo desta dissertação é melhorar o sistema de localização já existente do projeto SheepIT, descrito na secção 2.3. A solução proposta passa por melhorar as estivas das distâncias usadas na trilateração a partir do valor de RSSI obtido nas comunicações entre os faróis e as coleiras, sendo que para isso foram desenvolvidos 4 processos.

O primeiro consiste numa calibração do modelo usado no cálculo da distância, o segundo e terceiro consistem no desenvolvimento de dois filtros de estados e o quarto numa fusão sensorial do valor de distância estimada com valores de aceleração das coleiras utilizando um filtro de Kalman. O desenvolvimento e teste dos processos enunciados levou à necessidade de se criar diferentes datasets e para tal foi desenvolvido um simulador capaz de gerar os diferentes datasets, de aplicar os diferentes processos aos dados gerados e aplicar os diferentes algoritmos de localização. Para além disso o simulador tem a capacidade de mostrar graficamente todo o percurso e a localização atual das coleiras.

Quanto aos algoritmos de localização utilizados, estes permanecem os mesmos já apresentados anteriormente. Após a determinação da localização foi adicionado um outro processo com o objetivo de limitar a distância percorrida entre a última localização obtida e a localização atual. A arquitetura da solução proposta está representada na figura 4.1. A solução proposta começa por receber os valores de RSSI, aos quais é aplicado um filtro de estados. Com os valores de RSSI resultantes são estimada distâncias, de acordo com os parâmetros obtidos no processo de calibração. De seguida, às distâncias estimadas é aplicado um segundo filtro de estados ou um filtro de Kalaman. Com as distâncias filtradas e com algoritmos de trilateração é calculada uma posição preliminar, a qual passa por um conjunto de verificações, sendo ajustada se necessário.

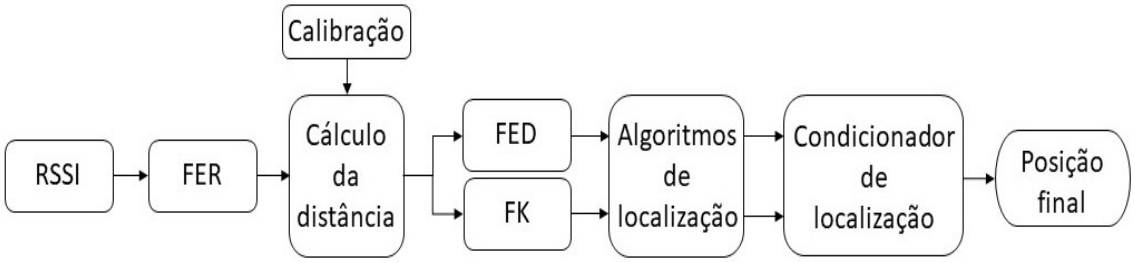

Figura 4.1: Arquitetura geral da solução proposta

4.2 Calibração

O mecanismo de localização animal existente foi desenvolvido numa altura em que o rádio usado nas comunicações entre os faróis e as coleiras era o CC1110 [30]. Este rádio permitia obter o valor de RSS em dBm sendo por isso usado o modelo *Log-normal shadow* para estimar a distância. Por questões de alcance, o rádio foi trocado para o RFM22B [31]. Este rádio, tal como o anterior, é capaz de indicar a intensidade do sinal recebido, mas com a diferença de que os valores fornecidos por este não são de potencia, mas sim indicativos de potencia, isto é, eles não têm unidade de medida e podem variar entre 0 e 255. Isto implica que o modelo usado anteriormente para estimar a distância não pode ser usado. Portanto foi necessário fazer um estudo da curva de variação do RSSI em função da distância de modo a conseguir-se extrair a equação 4.1. Esta equação é dada por uma regressão logarítmica de um conjunto de valores médios de RSSI de diferentes combinações coleira/farol, e resolvendo-a em ordem a d obtém-se a equação modelo (equação 4.2).

$$RSSI = A * ln(d) + B \tag{4.1}$$

$$d = e^{(\frac{RSSI-B}{A})} \tag{4.2}$$

Após esse estudo verificou-se que, apesar de se usar o mesmo modelo de rádio em todos os dispositivos, as curvas obtidas para cada uma da combinações coleira/farol eram muito semelhantes contendo todavia uma diferença visível de *offset* no parâmetro B da equação modelo, o que significa que para uma mesma distância são obtidos valores de RSSI diferentes para as varias combinações coleira/farol. A figura 4.2 é um exemplo ilustrativo das curvas obtidas para três combinações coleira/farol diferentes.

Como é possível observar na figura 4.2, para a mesma distância d, são obtidos três valores diferentes de RSSI ($rssi_1 < rssi_2 < rssi_3$). Aplicando estes valores de RSSI à equação modelo é possível observar que as estimativas de distância obtidas ($\widehat{d_1}, \widehat{d_2}, \widehat{d_3}$) estão afetados por um erro que é tanto maior quanto maior for a diferença do parâmetro B entre a curva média e a curva real, e que tanto pode ser positivo como negativo. Isto é, para os pares de coleira/farol que contêm uma curva cujo parâmetro B é superior ao parâmetro B da curva média (equação modelo) as distâncias estimadas pela equação modelo são sempre inferiores à distância real, sendo portanto para estes casos, cometido um erro negativo. Por outro lado, para os pares coleira/farol cujo o valor de B das suas curvas é inferior ao da curva média são estimadas distâncias superiores à distância real, o que leva a que seja cometido um erro positivo, com a agravante de este ser muito maior e de aumentar muito mais, com o aumento da distância, quando comparado ao caso anterior, como é possível ver na figura 4.2.

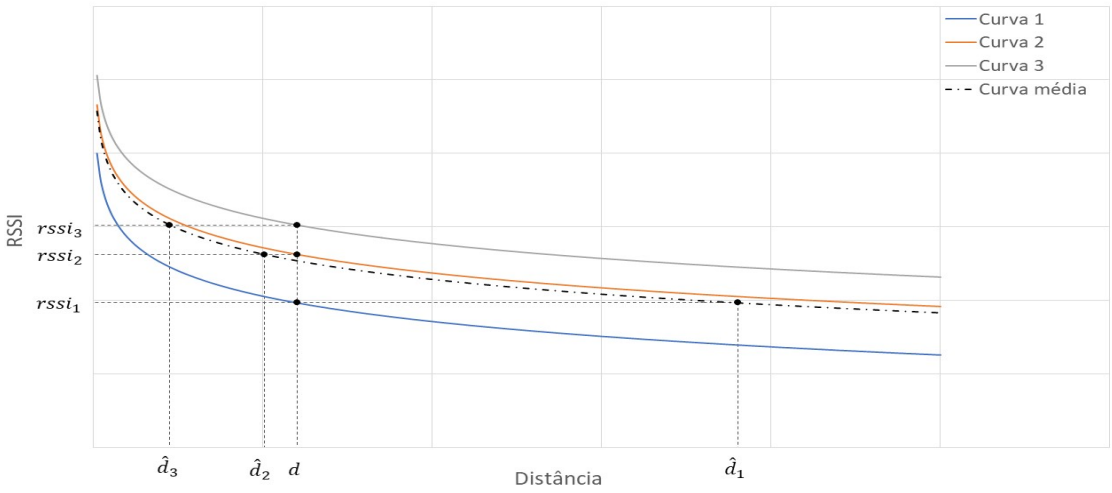

Figura 4.2: Diferença de *offset* das curvas

As diferenças do erro cometido pelo modelo no cálculo da distância levam, por si só, a que sejam estimadas distâncias com valores muito diferentes prejudicando a estimativa de localização do algoritmo de trilateração ou até mesmo, em alguns casos, causando a não interseção das circunferências da trilateração, sendo assim impossível achar uma localização (figura 4.3). Na figura 4.3 está representado um exemplo ilustrativo de três estimativas de distância diferentes para uma mesma distância real.

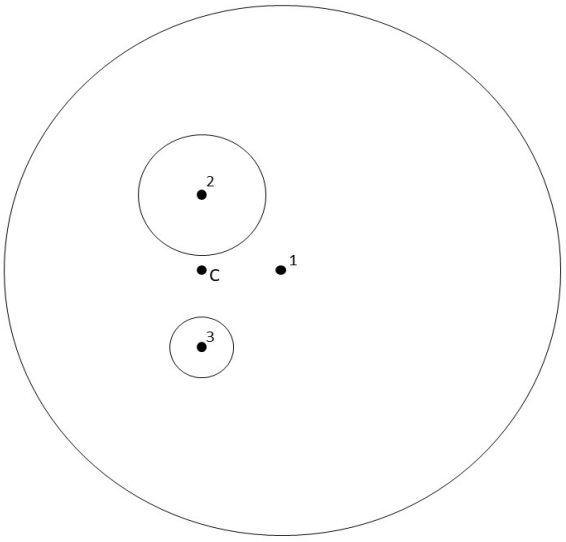

Figura 4.3: Não interseção das circunferencias

Assim para tentar homogeneizar o erro cometido no cálculo da distância, em vez de ser usado um modelo estático para calcular a distância, isto é, um modelo onde a equação usada para calcular a distância é a mesma para todas as combinações coleira/farol, optou-se por usar um modelo calibrado, tendo sido desenvolvido para tal um processo de calibração.

O processo da calibração passa então por determinar os valores do parâmetro B da equação modelo para todas as combinações de coleiras/faróis. Este processo é realizado apenas uma vez, antes de iniciar o sistema, e consiste em colocar todas as coleiras a uma distância pré-definida, medir o valor de RSSI médio de cada par coleira/farol para essa distância e calcular o parâmetro B, substituindo na equação 4.2 as variáveis de distância e RSSI pelos valores obtidos nas medições.

Por fim, os valores de calibração calculados são posteriormente passados para a gateway através de um ficheiro TXT que contém uma tabela de N coleiras por M faróis, e depois são guardados em memória numa matriz do mesmo tamanho.

4.3 Filtros de Estados

Como já foi referido anteriormente o sistema SheepIT permite saber o estado do animal. Este estado era inicialmente calculado na *gateway* utilizando os valores de aceleração dinâmica recebidos pela coleira, sendo obtido um de três estados possíveis (STOPPED, ACTIVE, RUNNING).

Atualmente o algoritmo que classifica o estado do animal está situado na coleira, sendo por isso enviado por esta aquando as comunicações entre as coleiras e os faróis. Para além do local onde é executado o algoritmo ter mudado, o número de estados possíveis também se alterou, passando assim a ser possível obter um dos seguintes quatro estados:

- STANDING - Animal encontra-se parado.
- EATING - Animal encontra-se a comer, podendo ou não haver uma pequena movimentação.
- MOVING - Animal encontra-se a andar.
- RUNNING - Animal encontra-se a correr.

Os filtros apresentados a seguir vão aplicar diferentes processos, consoante o estado recebido pela coleira, sendo que para ambos os filtros foi considerado que a movimentação que poderá existir no estado de EATING é mínima, levando assim a que o processamento seja igual para os estados de EATING e STANDING.

4.3.1 RSSI

O Filtro de Estados do RSSI (FER) tem como objetivo filtrar os valores de RSSI recebidos de acordo com o estado recebido da cada uma das coleiras de modo a que se obtenha uma melhor estimativa da distância. Assim a ideia do filtro passa por tentar melhorar os valores do RSSI quando o estado atual da coleira caracteriza uma movimentação mínima ou nula.

Como é possível observar no Algoritmo 1, o filtro começa por identificar o estado atual da coleira e, caso este seja de STANDING ou EATING, o valor de RSSI usado para o cálculo da distância é dado pela média dos últimos valores de RSSI consecutivos cujos estados também sejam STANDING ou EATING. No caso do estado atual ser MOVING ou RUNNING o valor de RSSI a ser usado é dado pelo valor de RSSI recebido pela coleira. No algoritmo o *BID* representa o identificador do farol que devolveu o RSSI.

Algoritmo 1: Filtro de estados do RSSI

```
1: function FER(RSSI, BID, state)
2:     if state == STANDING||state == EATING then
3:         for i = 1 : N_RSSI do
4:             RSSI_count[BID]++
5:             RSSI_sum[BID]+=RSSI
6:             RSSI_final[BID] = RSSI_sum/RSSI_count
7:     else
8:         for i = 1 : N_BEACONS do
9:             RSSI_count[BID]=0
10:            RSSI_sum[BID]=0
11:            RSSI_final[BID] = RSSI
       return RSSI_final
```

4.3.2 Distância

À semelhança do anterior, Filtro de Estados da Distância (FED) tem como objetivo melhorar os valores de distância estimados quando o estado atual da coleira é caracterizado por uma movimentação mínima ou nula.

Como é possível observar no Algoritmo 2, este filtro começa por executar o mesmo processamento que o filtro anterior, com a diferença de que este em vez de ser aplicado aos valores de RSSI recebidos pelas coleiras é aplicado diretamente nas distâncias estimadas a partir desses valores de RSSI.

Uma vez que este filtro trabalha com valores de distâncias, para o caso de o estado atual ser MOVING ou RUNNING foi adicionado um segundo processo de filtragem que tem como objetivo controlar a variação entre duas distâncias consecutivas estimadas para um mesmo farol. Este processo levou à necessidade de se manter um histórico das últimas distâncias estimadas e dos seus respetivos identificadores dos faróis, de modo a que para cada uma das distâncias atuais seja obtido o seu identificador do farol (BID) correspondente e comparado com os identificadores recebidos na última iteração. Assim, as distâncias atuais cujos respetivos identificadores não estejam contidos no histórico não sofrem qualquer alteração por parte deste processo. Por outro lado, para aquelas cujos identificadores tenham sido recebidos na iteração anterior e cuja a variação das distâncias seja superior a d_{max}, é considerado que o animal no máximo teve uma deslocação, em relação ao farol, caracterizada por uma das duas hipóteses apresentadas na figura 4.4.

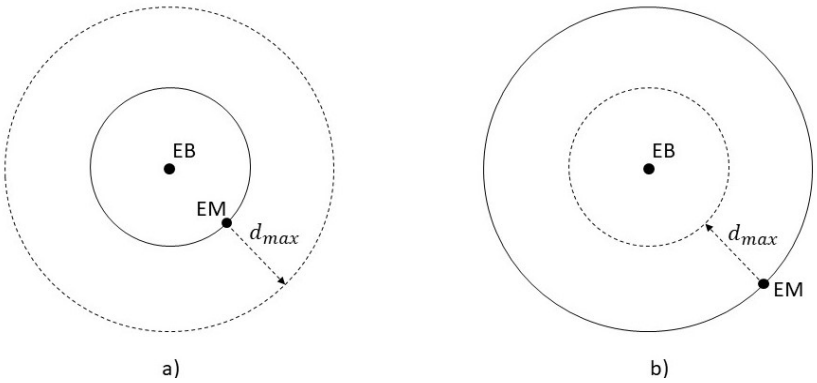

Figura 4.4: Casos de deslocação máxima em relação a um determinado farol

A alínea a) da figura 4.4 representa então a deslocação máxima do animal quando este se afasta do farol, enquanto a alínea b) representa a deslocação máxima quando o animal se aproxima do farol e d_{max} representa a variação máxima entre duas distâncias consecutivas. O valor de d_{max} é calculado partindo do pressuposto que o animal teve um deslocamento com uma velocidade constante durante o intervalo de tempo entre a medição das distâncias, sendo dado por:

$$d_{max} = V_{state} * \Delta t \tag{4.3}$$

onde Δt corresponde ao intervalo de tempo e V_{state} é a velocidade a que o animal se moveu sendo o valor desta depende do estado atual registado.

A escolha da deslocação máxima a ser usada no limitador de distância do filtro é feita comparando as distâncias calculadas da iteração anterior e as atuais, ou seja, se a distância ao farol atual for maior que a distância anterior calculada então o filtro escolhe o caso a) sendo a distância estimada dada pela soma da última distância recebida com d_{max}. Por outro lado, se a distância atual calculada for menor que a distância anterior então é escolhido o caso da alínea b) e a distância estimada passa a ser dada pela subtração da última distância recebida com d_{max}.

4.4 Filtro de Kalman

Para que seja possível a aplicação de um filtro de Kalman é necessário, em primeiro lugar, descrever o sistema num modelo de espaço de estados de acordo com as seguintes equações:

Equação de estado:

$$x_k = Ax_{k-1} + Bu_k + \varepsilon_x \tag{4.4}$$

Equação de saída:

$$y_k = Hx_k + \varepsilon_z \tag{4.5}$$

onde, A, B e H são as matrizes correspondentes do algoritmo, x_k é o vetor das variáveis de estado do sistema, u_k é uma entrada conhecida do sistema, y_k é o valor de saída, ε_x é o ruído do processo e ε_z é o ruído de medição.

Nesse sentido, como o objetivo da solução proposta passa por tentar melhorar as estimativas da distância usadas no cálculo da localização foi considerado que para cada par coleira/farol a coleira tem um movimento uniformemente variado em relação a esse farol, levando assim a que seja aplicado um filtro de Kalman com o mesmo modelo de espaço de estados para cada

Algoritmo 2: Filtro de estados da Distância

```
1: function FER(RSSI, BID, state)
2:     if state == STANDING||state == EATING then
3:         for i = 1 : N_RSSI do
4:             Dist_count[BID]++
5:             Dist_sum[BID]+=Dist
6:             Dist_final[BID] = Dist_sum[BID]/Dist_count[BID]
7:     else
8:         for i = 1 : N_BEACONS do
9:             Dist_count[BID]=0
10:            Dist_sum[BID]=0
11:            Dist_final[BID] = Dist
12:        if state == MOVING then
13:            d_max = V_MOVING*dt
14:        else
15:            d_max = V_RUNNING*dt
16:        for i = 1 : N_RSSI do
17:            if BID received in previous iteration then
18:                if Dist_final[BID] - Last_Dist[BID] > d_max then
19:                    Dist_final[BID] = Last_Dist[BID] +d_max
20:                if Dist_final[BID] - Last_Dist[BID] < -d_max then
21:                    Dist_final[BID] = Last_Dist[BID] - d_max
       return Dist_final
```

um dos pares coleira/farol existentes. Assim, para descrever o modelo de espaço de estados do sistema foram usadas as seguintes equações do movimento uniformemente variado:

Equação da posição

$$p_k = p_{k-1} + v_{k-1}t + \frac{1}{2}a_k t^2 \tag{4.6}$$

Equação da velocidade

$$v_k = v_{k-1} + a_k t \tag{4.7}$$

Posto isto é possível definir o vetor de estados do sistema, que é composto pela posição e velocidade:

$$x_k = \begin{bmatrix} p_k \\ v_k \end{bmatrix} \tag{4.8}$$

Quanto à entrada do sistema, u_k, esta é dada pela aceleração dinâmica da coleira ou mais concretamente pela magnitude da aceleração dinâmica. Apesar de ser possível obter a aceleração dinâmica para os eixos x, y e z, optou-se por calcular a magnitude da aceleração usando apenas os valores dos eixos das abcissas e das ordenadas pois considerou-se que a localização é feita num plano sem inclinação. Temos, portanto, que:

$$u_k = [a_k] \tag{4.9}$$

$$a_k = \sqrt{a_{xk}^2 + a_{yk}^2} \tag{4.10}$$

Assim, o modelo de espaço de estados do sistema pode ser descrito a partir das seguintes equações:

$$x_k = \begin{bmatrix} 1 & t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} p_{k-1} \\ v_{k-1} \end{bmatrix} + \begin{bmatrix} \frac{t^2}{2} \\ t \end{bmatrix} a_k + \varepsilon_x \tag{4.11}$$

$$y_k = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} p_k \\ v_k \end{bmatrix} + \varepsilon_z \tag{4.12}$$

Onde,

$$A = \begin{bmatrix} 1 & t \\ 0 & 1 \end{bmatrix} \tag{4.13}$$

$$B = \begin{bmatrix} \frac{t^2}{2} \\ t \end{bmatrix} \tag{4.14}$$

$$C = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{4.15}$$

Após isto é necessário definir os parâmetros do filtro correspondentes ao ruído, matrizes Q e R. O filtro de Kalman requer que estes parâmetros sejam definidos em matrizes de covariância do ruído. Assim temos,

$$Q = \begin{bmatrix} \sigma_p^2 & \sigma_p \sigma_v \\ \sigma_v \sigma_p & \sigma_v^2 \end{bmatrix} \tag{4.16}$$

$$R = \begin{bmatrix} \sigma_z^2 \end{bmatrix} \tag{4.17}$$

onde σ_p^2 e σ_v^2 correspondem as variâncias da posição e da velocidade respetivamente e σ_z^2 é a variância da medição. A partir da equação (4.11) é possível ver que a variância da posição e da velocidade são dadas pela variância da aceleração, σ_a^2, e por isso Q fica igual a:

$$Q = \sigma_a^2 \begin{bmatrix} \frac{t^4}{4} & \frac{t^3}{2} \\ \frac{t^3}{2} & t^2 \end{bmatrix} \tag{4.18}$$

Para executar o filtro é também necessário definir algumas condições iniciais, que são dadas pelas variáveis $\widehat{x}_{k-1}$ e P_{k-1}. A variável $\widehat{x}_{k-1}$ é inicializada com o valor de velocidade $v_{k-1} = 0(m/s)$ e com a posição p_{k-1} dada pelo valor da primeira estimativa da distância. Uma vez que os valores com que as variáveis anteriores foram inicializadas não são conhecidos com precisão, então a variável P_{k-1} é inicializada com uma matriz diagonal de um valor alto:

$$P_{k-1} = \begin{bmatrix} 200 & 0 \\ 0 & 200 \end{bmatrix} \tag{4.19}$$

Como foi referido no capítulo anterior o filtro de Kalman pode ser usado em tempo real requerendo apenas as medições presentes e os valores das variáveis de estado da iteração anterior. Isto implica que para manter os valores das variáveis de estado anteriores válidos a qualquer instante de tempo, era necessário que a cada iteração se calculasse a distância a que a coleira se encontra de todos os faróis. No entanto, o projeto SheepIT está desenhado para receber na gateway apenas quatro valores de RSSI por cada coleira. Assim, se no sistema existirem mais do que 4 faróis não é possível calcular a distância a que a coleira de encontra de todos os faróis a cada iteração pois à medida que a coleira se movimenta esta irá aproximar-se

de uns faróis e afastar-se de outros, levando a casos onde o RSSI de determinados faróis deixa de ser recebido e se passa a receber o de outros.

Em consequência disto, quando o RSSI de um farol volta a ser recebido as últimas variáveis de estado calculadas para esse par coleira/farol poderão já estar desatualizadas, levando à necessidade de uma reinicialização das condições iniciais. Contudo quando um farol deixa de pertencer e depois volta a pertencer ao conjunto do 4 recebidos podemos estar perante dois casos diferentes cuja reinicialização das condições iniciais pode ser ou não necessária. O primeiro caso, em que é necessária a reinicialização, consiste quando a coleira se movimentou para uma zona onde existiam 4 faróis mais próximos do que aquele que deixou de ser ouvido, se manteve por lá durante algum tempo e depois voltou a movimentar-se para a zona onde estava inicialmente. O segundo caso, onde não é necessária a reinicialização, é o caso em que a coleira se está a movimentar numa zona, à qual se deu o nome de zona de transição, que corresponde a uma zona onde, apesar da coleira estar mais perto de um farol A, as flutuações do próprio RSSI fazem com que o farol A pareça estar mais longe do que um farol B fazendo com que o RSSI recebido seja o do farol B em vez do farol A.

Para escolher se estamos no primeiro ou no segundo casos apresentados, foi desenvolvido um mecanismo onde são contadas o número de ausências para cada par coleira/farol e, desta forma, quando se volta a receber o RSSI de um farol, é comparado o número de ausências com um valor limite N. Se o número de ausências for maior que N então é considerado o primeiro caso. Caso seja menor ou igual a N então é considerado o segundo caso. O valor de N escolhido foi 3 visto que o sistema SheepIT está desenhado para receber na gateway novos valores com um intervalo de tempo de 6 segundos, o que faz com que sejam esperados 24 segundos até reiniciarem as condições iniciais.

4.5 Condicionador de localização

Após ser executado o algoritmo de localização são verificadas algumas condições que vão influenciar a posição final da coleira.

Em primeiro lugar é verificado se foi recebido algum valor de RSSI cuja distância filtrada correspondente é menor que um valor *thr1*, que para o efeito foi definido como 5m, distância correspondente ao *thr1* do algoritmo avançado de localização apresentado na secção 2.3. Caso esta condição se verifique, então a posição final da coleira é dada pela posição do farol que devolveu esse valor de RSSI.

No caso desta primeira não se confirmar é verificado se a posição estimada pelo algoritmo de localização é valida. Caso seja valida, analogamente ao que acontece no filtro FER, é também aplicado um limitador de distância percorrida, depois da localização obtida pelos algoritmos de localização. Este limitador baseia-se também no estado recebido para determinar a distância máxima entre a duas localizações consecutivas. Assim é calculada a distância entre a última localização e a atual obtida pelo algoritmo de localização, e caso esta seja maior do que a distância máxima calculada, então é calculado um ângulo de movimentação θ e as novas coordenadas da localização atual são dados por:

$$\begin{aligned} Pos[t].x &= Pos[t-1].x + d_{max} * cos(\theta) \\ Pos[t].y &= Pos[t-1].y + d_{max} * sin(\theta) \end{aligned} \tag{4.20}$$

onde $Pos[t]$ e $Pos[t-1]$ correspondem à posição atual e anterior, respetivamente, da coleira.

Por outro lado, caso a posição estimada pelo algoritmo de localização não seja valida, a posição final da coleira é dada pela última posição final estimada dessa mesma coleira.

4.6 Discussão

Neste capítulo é descrita a arquitetura e implementação do sistema de localização do projeto SheepIT. Foram desenvolvidos quatro novos processos, um primeiro que consiste numa calibração do modelo de propagação usado no calculo das distâncias devido às diferença de *hardware* do novo rádio usado para as comunicações. Os outros três processos consistem em mecanismos de filtragem com o objetivo de minimizarem o erro nas distâncias estimadas para a localização. Dois dos mecanismos de filtragem baseiam-se no estado devolvido pela coleira para aplicarem diferentes processamentos, sendo que um deles é aplicado aos valores de RSSI e o outro às distâncias estimadas. O terceiro mecanismo consiste um filtro de Kalman onde é feito *sensor fusion* das distâncias estimadas com dados de aceleração. Todos estes processos foram testados e analisados no Capítulo 5.

Capítulo 5

Testes e verificação

Nesta secção são apresentados e analisados os resultados obtidos nos testes experimentais efetuados ao sistema de localização. Num primeiro momento é apresentado um estudo da variação do RSSI em função da distância para o rádio utilizado nas comunicações seguido da análise do processo de calibração do modelo usado para o cálculo da distância. Posteriormente é apresentada a análise dos testes efetuados aos filtros desenvolvidos em conjunto com os algoritmos de localização. Por fim é apresentada uma análise relativa à utilização de recursos para cada um dos mecanismos desenvolvidos.

5.1 RSSI

Como foi referido no capítulo anterior, o rádio usado nas comunicações onde se obtêm os valores de RSSI usados no cálculo das distâncias foi trocado e, portanto, foi necessário obter outro modelo para o cálculo da distância. Este modelo foi obtido experimentalmente através de um estudo da variação do RSSI com a distância.

Para isso foi realizado um teste que consistiu em se colocar 2 faróis lado a lado e medir o RSSI de 5 coleiras todas a mesma distância dos faróis. Numa perspetiva de se tentar imitar as condições reais, do funcionamento do sistema, os faróis foram colocados a uma altura de 3m e as coleiras foram colocadas numa estrutura com cerca de 55cm de altura, que corresponde à altura média das ovelhas medida na Escola Superior Agrária de Viseu - Instituto Politécnico de Viseu. Os valores de RSSI foram medidos de 3 em 3 metros para distâncias entre os 3m e os 210m pois as linhas de videiras na Quinta do Pato também estão distanciadas de 3m umas das outras. Para cada uma dessas distâncias foi feita a média de 30 valores de RSSI de modo a se obter um valor mais próximo do real. Na figura 5.1 estão representados os valores medidos para as diferente coleiras.

Pela análise da figura 5.1 e da tabela 5.1 é possível observar que as curvas das diferentes coleiras são muito parecidas contendo apenas um deslocamento visível entre elas. Esta diferença nas curvas pode ser explicada pelas diferenças que podem ocorrer no fabrico do *hardware*.

De forma a se obter uma equação modelo foi desenhada uma nova curva correspondente aos valores médios do RSSI das diferentes coleiras (figura 5.2). A par desta foram desenhadas outras duas curvas correspondentes aos valores máximos e mínimos de RSSI obtidos nas diferentes coleiras.

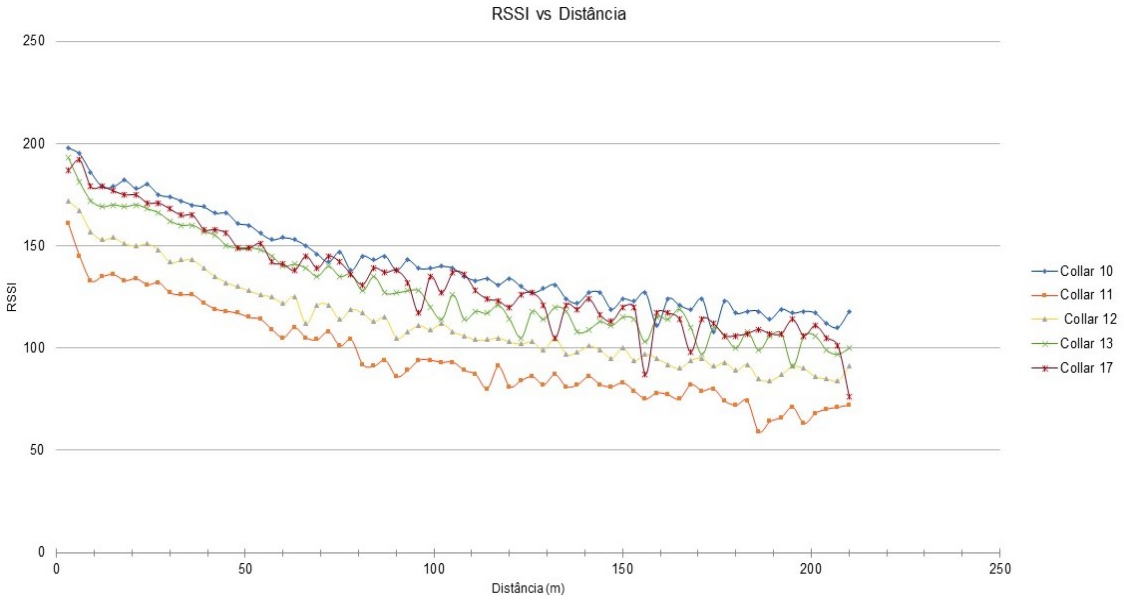

Figura 5.1: Valores médios de RSSI para cada coleira

Tabela 5.1: Regressão logarítmica da variação do RSSI em função da distância para cada coleira

Coleira	**Regressão logaritmica**
Coleira 10	rssi = -24.59*ln(d) + 249.17
Coleira 11	rssi = -24.51*ln(d) + 203.45
Coleira 12	rssi = -24.26*ln(d) + 219.41
Coleira 13	rssi = -25.69*ln(d) + 241.59
Coleira 17	rssi = -26.12*ln(d) + 247.60

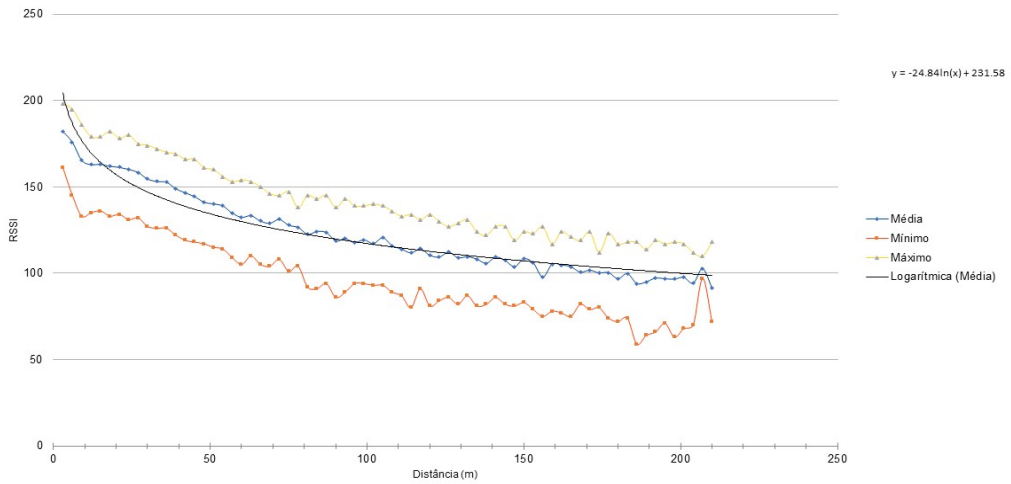

Figura 5.2: Valor médio, máximo e mínimo de RSSI das diferentes coleiras

A equação 5.1 foi obtida através da regressão logarítmica da curva média.

$$RSSI = -24.84 * ln(d) + 231.58 \tag{5.1}$$

Uma vez que o objetivo é determinar a distância, resolveu-se a equação 5.1 em ordem a d obtendo-se a equação modelo (5.2).

$$d = e^{(\frac{231.58 - RSSI}{24.84})} \tag{5.2}$$

Para avaliar a dispersão das estimativas de distância dadas pela equação 5.2 foram aplicados à equação modelo valores de RSSI, obtidos através das regressões logarítmicas dos diferente pares coleira/farol (tabela 5.1), correspondentes as distância de 50m e 100m. Os resultados obtidos são apresentados na tabela 5.2 e é possível constatar que de facto a dispersão dos valores de distância estimados é muito elevada. Os valores de distância estimada são mínimos para a coleira 10 e correspondem a aproximadamente metade do valor da distância real. Por outro lado, os valores de distância estimada são máximos para a coleira 11 e correspondem aproximadamente ao triplo do valor da distância real. Isto significa que à medida que a distância real aumenta a dispersão dos valores de distância estimada também aumenta levando assim a que seja cometido um erro grosseiro na localização das coleiras.

Tabela 5.2: Dispersão dos valores de distância estimada para os diferentes pares coleira/farol

Distância Real (m)	**Distância estimada (m)**				
	Coleira 10	**Coleira 11**	**Coleira 12**	**Coleira 13**	**Coleira 17**
50	23.65	150.70	76.01	38.34	32.64
100	46.89	298.77	150.70	79.14	67.37

Assim é possível concluir que o uso de uma só equação modelo para todos os pares coleira/farol não corresponde a uma solução viável. No sentido de contornar este problema de dispersão de valores optou-se por calibrar a equação modelo para cada um dos pares coleira/farol, obtendo-se desta forma a nova equação modelo:

$$d = e^{(\frac{B - RSSI}{24.84})} \tag{5.3}$$

onde B é o parâmetro obtido pelo processo de calibração.

5.2 Calibração

Como referido no capítulo anterior, o processo de calibração consiste em achar um valor para o parâmetro B da equação modelo para cada um dos pares coleira/farol. O cálculo destes valores é feito através de medições do RSSI nas comunicações entre as coleiras e os faróis a uma distância real conhecida. Este valores, por sua vez, são aplicados à equação 5.4, que deriva da equação 5.3, e são obtidos os valores do parâmetro B.

$$B = RSSI + 24.84 * ln(d) \tag{5.4}$$

Para escolher a distância a que deve ser realizada a calibração foi considerado que as regressões logarítmicas da tabela 5.1 representam as curvas ideais para cada par coleira/farol, ou seja, que o objetivo é calcular um parâmetro B para o qual a equação modelo se aproxime o

mais possível de cada uma dessas regressões. Nesse sentido, começou-se por calcular a média da diferença entre os valores médios de RSSI apresentados na figura 5.1 e a suas regressões logarítmicas para todas as distâncias por forma a encontrar um distância onde essa diferença seja menor, obtendo-se a figura 5.3. É possível observar que esta começa por apresentar uma diferença negativa, isto é, os valores de RSSI medidos experimentalmente são menores do que os valores dados pelas regressões, e que entre os 15m e os 18m passa de negativa a positiva. Este fenómeno acontece devido à diferença de alturas entre as antenas dos faróis e das coleiras que dificulta a propagação do sinal. Apesar do valor da diferença nesta zona passar por zero é possível observar que uma pequena variação da distância real pode fazer com que a diferença dos valores de RSSI seja grande. Continuando a análise do gráfico é percetível que existe uma outra zona, situada por volta dos 100m, onde o valor desta diferença passa de positiva para negativa. Ao contrário do que acontece na zona dos 15m aos 18m, nesta zona é possível observar que o valor da diferença é muito próximo de zero para uma maior gama de valores de distância real, fazendo assim com que um pequeno desvio na distância real não causa uma diferença acentuada nos valores de RSSI. Posto isto, a distância escolhida para o processo de calibração é de 100m.

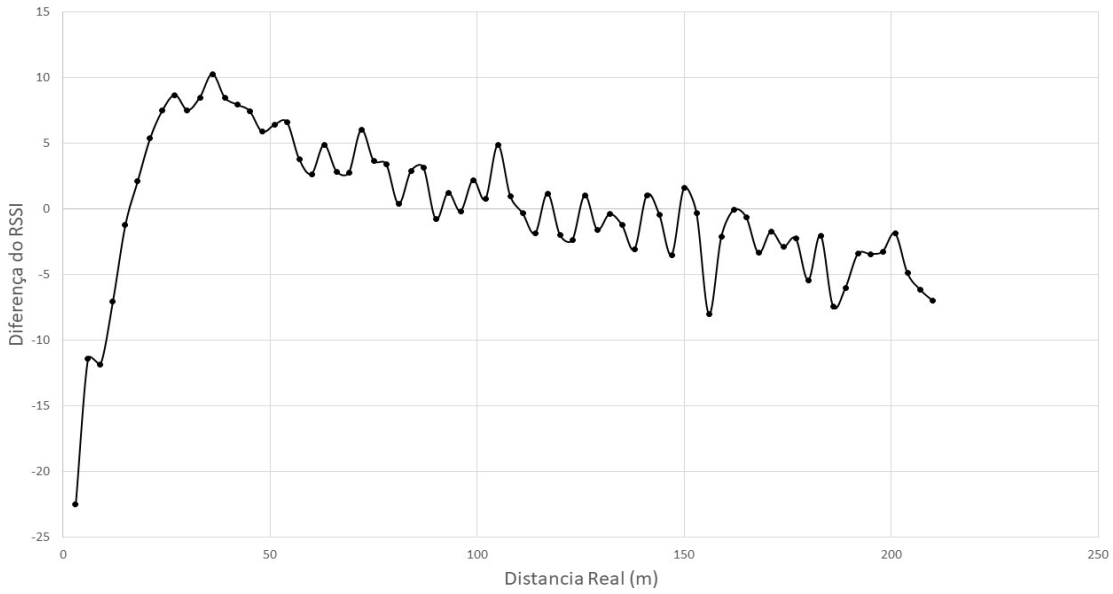

Figura 5.3: Diferença do RSSI entre valores medidos experimentalmente e a regressão logaritmica dos mesmos

Para validar o modelo calibrado procedeu-se então a realização de uma experiência que consistiu em recolher valores de RSSI entre três coleiras e dois faróis, para as distâncias de 25, 50, 75, 100 e 125 metros. Esta experiência foi realizada em dois dias diferentes de modo a garantir se existe repetibilidade dos resultados. Uma vez obtidos os valores de RSSI procedeu-se à calibração do modelo com base nos valores de RSSI obtidos para as distâncias de 100m no primeiro dia. Após isto, foram estimadas as distâncias correspondentes aos valores de RSSI medidos nos dois dias, usando o modelo calibrado. Para além destas também foram estimadas distâncias usando o modelo estático por forma a comparar os valores estimados. Os gráficos das figuras 5.4 e 5.5 apresentam os resultados obtidos nos dois dias, para um dos faróis.

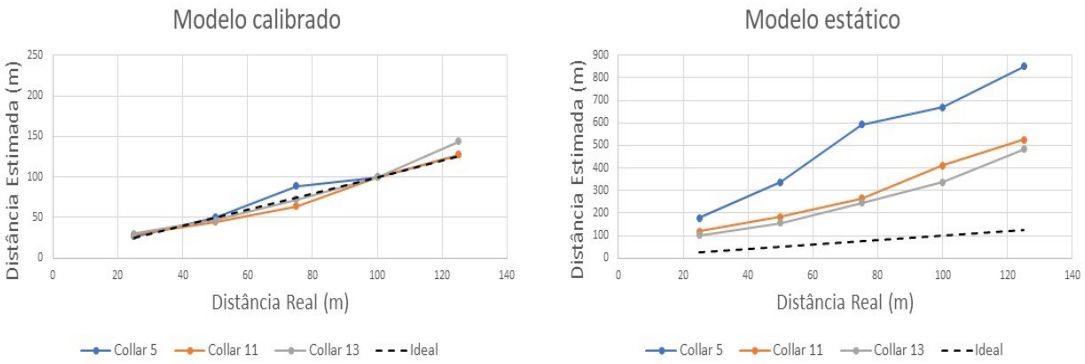

Figura 5.4: Distâncias estimadas com os valores do primeiro dia

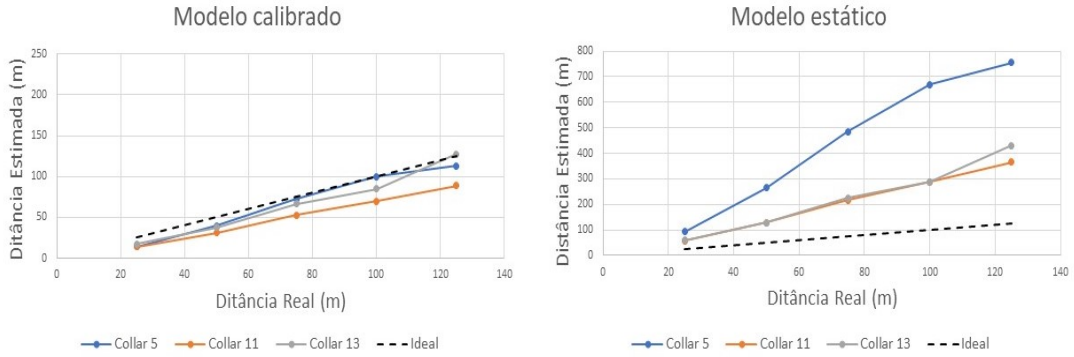

Figura 5.5: Distâncias estimadas com os valores do segundo dia

Pela análise das figuras é possível concluir que o uso do modelo calibrado para estimar as distâncias diminui a dispersão dos valores estimados para os diferentes pares coleira/farol quando comparado com o modelo estático que era usado inicialmente. Na figura 5.6 está representada a dispersão máxima obtida pelos dois modelos para cada uma das distâncias dos dois dias e, como é possível observar, os valores de dispersão obtidos pelo modelo calibrado são aproximadamente dez vezes mais pequenos que os obtidos pelo modelo estático.

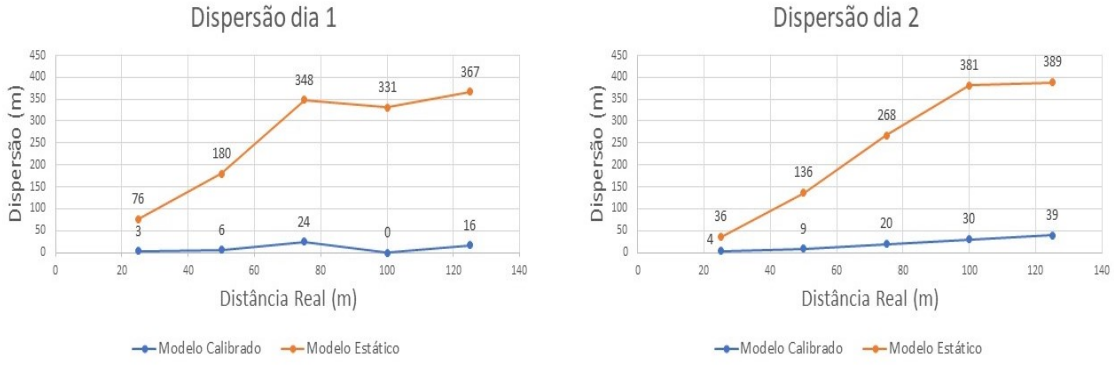

Figura 5.6: Dispersão máxima das estimativas da distância

5.3 Avaliação dos mecanismos de filtragem de erro

Para analisar o comportamento dos filtros e dos algoritmos de localização foram realizadas simulações onde se calculou a localização de uma coleira para diferentes fases de filtragem. A figura 5.7 ilustra as diferentes fases para as quais se calculou a localização da coleira.

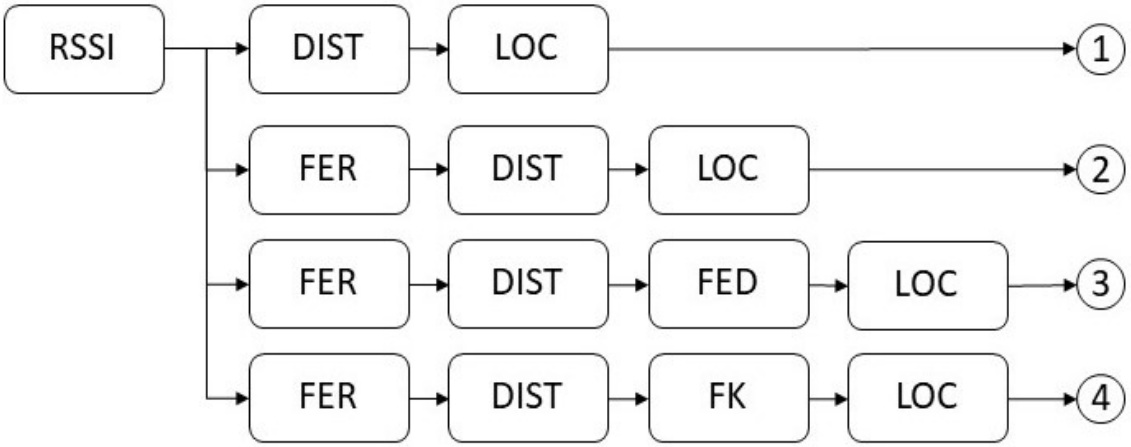

Figura 5.7: Diagrama das fases de filtragem

Na figura 5.7 o ponto 1 corresponde a localização obtida com os dados não tratados, o ponto 2 à localização obtida só com o filtro FER, o ponto 3 a localização obtida no conjunto do filtro FER com o Filtro de Kalman e o ponto 4 à localização obtida com o filtro FER e o filtro FED. Para cada uma destas fases foi calculada a localização com o algoritmo de distâncias mais próxima e com o algoritmo de interseção de linhas. Para além disso foi obtida também uma terceira localização, que é dada pelo ponto médio das localizações obtidas com os dois algoritmos anteriores.

Numa primeira parte estas simulações foram feitas com dados sintetizados artificialmente, tendo sido depois feitas com dados obtidos num ambiente real (vinha da Adega Luís Pato).

5.3.1 Sintetizador de dados

As simulações feitas com os dados sintetizados foram efetuadas para um circuito de 16 faróis com uma configuração de 4x4, espaçados de 50m, e para um percurso em forma de zig-zag que passava entra cada uma das fileiras dos faróis. Uma vez definido o percurso foi necessário desenvolver um algoritmo capaz de simular a movimentação aleatória de uma coleira ao logo desse percurso de forma a gerar os dados necessários aos filtros, mais concretamente valores de RSSI, estados da coleira e valores de aceleração. Apesar de no total poderem existir 4 estados possíveis, para simplificar a geração dos dados o algoritmo apenas gera dados com 3 desses estados, estado RUNNING, MOVING e STANDING, visto que o processamento dos filtros de estados é o mesmo para os estados STANDING e EATING.

Posto isto, o algoritmo começa por gerar as posições virtuais da coleira ao mesmo tempo que gera o estado para cada uma das posições. Para isso, por uma questão de simplicidade, é gerado um número entre 0 e 1 com uma probabilidade de 75% de gerar o número 1 e com 25% de gerar o número 0 de modo a ser gerado um estado com uma probabilidade maior que outro. Caso o número gerado seja igual a 0, o algoritmo gera o estado RUNNING e a posição real

da coleira avança 2m no percurso definido. Caso o número gerado seja igual a 1, o algoritmo gera o estado MOVING e a posição real da coleira avança 1m no percurso. Neste último caso, depois da posição real da coleira ser atualizada, é ainda determinado se nas iterações seguintes a coleira se vai encontrar no estado STANDING e, em caso positivo, durante quantas iterações a coleira se manterá neste estado. Para isso, é gerado um segundo número com as mesmas características do primeiro. Caso este seja igual a 0 então é considerado que a coleira se encontrará no estado STANDING e é gerado um número aleatório entre 1 e 6 correspondente ao número de iterações em que a coleira se encontrará neste estado, de modo a se conseguir gerar vários estados STANDING consecutivos para testar os filtros de estados. A posição real da coleira para este estado é dada pela última posição real atualizada. Este processo é repetido até ao fim do percurso definido.

Para se gerar os valores de aceleração considerou-se que o intervalo de tempo, Δt, entre cada uma das posições reais geradas era de 1 segundo. Desta forma o algoritmo calcula a aceleração através das equações 5.5 e 5.6, onde Δd corresponde a distância percorrida entre duas posições reais consecutivas.

$$v(t) = \frac{\Delta d}{\Delta t} \tag{5.5}$$

$$a(t) = \frac{v(t) - v(t-1)}{\Delta t} \tag{5.6}$$

A estes valores de aceleração foi adicionado ruído com uma distribuição gaussiana centrado em zero e com um desvio padrão σ_a. Uma vez que a aceleração máxima é de $2m/s^2$, o desvio padrão σ_a definido é igual a $0.2m/s^2$.

Para gerar os valores de RSSI a serem usados nos testes o algoritmo começa por calcular para cada uma das posições reais da coleira a distância entre essa mesma posição e a posição de todos os faróis. Uma vez obtidas as distâncias a que a coleira se encontra de todos os faróis, estas distâncias são convertidas em valores de RSSI através da equação 5.2. De seguida é gerado um valor de erro aleatório que pertence ao intervalo $[-\frac{ERRO_{RSSI}}{2}, \frac{ERRO_{RSSI}}{2}]$, que é arredondado e posteriormente adicionado aos valores de RSSI calculados. Assim, tanto para o $ERRO_{RSSI} = 3$ como $ERRO_{RSSI} = 4$ pode ser gerado um erro máximo de 4 unidades de RSSI, sendo que a probabilidade de isso acontecer é menor para o $ERRO_{RSSI} = 3$. As coleiras do sistema SheepIT estão programadas para enviar apenas os quatro melhores valores de RSSI que obtêm durante as comunicações, isto é, na fase de sincronização as coleiras obtêm um valor de RSSI para cada um dos faróis sendo depois escolhidos os quatro maiores valores de RSSI e enviados para a gateway. Por isso, à semelhança do que acontece com as coleiras para cada uma das posições reais, o algoritmo, escolhe os quatro maiores valores de RSSI.

5.3.2 Resultados com dados simulados

Para analisar o comportamento dos filtros e dos algoritmos de localização realizaram-se 100 simulações para cada um dos valores do $ERRO_{RSSI}$ e foi feita uma média do erro médio de localização e da distância percorrida dessas simulações.

Assim, os resultados obtidos para os dados não tratados estão na tabela 5.3.

Tabela 5.3: Erro de localização dos dados não tratados

Erro de localização com os dados não tratados (m)									
Alg. $\backslash ERRO_{RSSI}$	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Distância próx.	6.22	7.61	9.19	11.14	13.39	15.88	18.30	20.74	23.12
Interseção linhas	5.53	7.60	9.73	12.02	14.36	16.75	19.24	21.73	24.14
Média dos 2 ant.	5.17	6.81	8.57	10.59	12.83	15.25	17.69	20.13	22.49

Filtro de Estados do RSSI

Tabela 5.4: Erro de localização do Filtro de Estados do RSSI

Erro de localização do Filtro de Estados do RSSI (m)									
Alg. $\backslash ERRO_{RSSI}$	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Distância próx.	4.16	5.25	6.40	7.80	9.54	11.29	13.51	15.51	17.71
Interseção linhas	3.44	4.78	6.32	8.22	10.21	12.30	14.73	16.88	19.25
Média dos 2 ant.	3.55	4.84	6.20	7.85	9.72	11.64	13.97	16.07	18.35

Filtro de Estados da Distância

Tabela 5.5: Erro de localização do Filtro de Estados da Distância

Erro de localização do Filtro de Estados da Distância (m)									
Alg. $\backslash ERRO_{RSSI}$	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Distância próx.	4.70	5.81	6.91	8.09	9.57	10.93	12.82	14.78	16.72
Interseção linhas	3.53	4.79	6.24	7.97	9.90	11.83	14.21	16.30	18.50
Média dos 2 ant.	3.82	5.06	6.35	7.81	9.51	11.16	13.32	15.36	17.44

Pela análise das tabelas 5.3 e 5.4 é possível concluir que existe uma redução do erro médio de localização quando é aplicado o filtro de estados do RSSI para ambos os algoritmos de localização. Esta redução encontra-se na ordem dos 30% aos 40% para os valores de $ERRO_{RSSI}$ mais baixos, diminuído até aos 20% para os valores de $ERRO_{RSSI}$ mais elevados.

Analisando também as tabelas 5.4 e 5.5 é possível concluir que apenas para os valores de $ERRO_{RSSI}$ maiores é que a aplicação do filtro de estados da distância surte efeito.

É possível reparar também que para valores de $ERRO_{RSSI}$ mais pequenos o algoritmo de interseção de linhas obtém melhores valores de erro de localização do que o algoritmo de distância mais próxima e que este comportamento se inverte para os valores de $ERRO_{RSSI}$ maiores. Contudo, não foi possível encontrar uma explicação para este comportamento sendo, portanto, necessário que se efetuem mais testes aos algoritmos de trilateração de modo a perceber o porquê deste comportamento.

Filtro de Kalman

Para o filtro de Kalman foi necessário variar o parâmetro referente ao erro da distância, σ_z, para cada um dos diferentes valores de $Erro_{RSSI}$ de modo a encontrar o valor que melhor

resultados apresenta. Isto deve-se ao facto de o erro gerado ser aplicado no valor de RSSI e do valor da distância correspondente a este RSSI ser dado por uma função exponencial fazendo com que o erro dependa da distância real a que se encontra. Uma vez que o erro máximo adicionado aos valores de RSSI é de ±5 ($ERRO_{RSSI} = 10$), o parâmetro σ_z foi testado para os valores: 0.05, 0.1, 0.15, 0.20, 0.30, 0.40, 0.50, 0.60, 0.70, 0.80, 0.90, 1.00, 2.00, 3.00, 4.00, 5.00, 6.00, 7.00, 8.00 e 9.00.

Na tabela 5.6 são apresentados os valores de σ_z para os quais se obteve um erro de localização menor e na tabela 5.7 são apresentados esses mesmos valores de erro. Os restantes valores de erro obtidos para os diferentes valores de σ_z encontram-se nas tabelas do Anexo A.

Tabela 5.6: Melhores valores do parâmetro σ_z do Filtro de Kalman

Melhores parâmetros do Filtro de Kalman									
Alg. \ $ERRO_{RSSI}$	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Distância próx.	1.00	1.00	0.80	2.00	2.00	3.00	3.00	4.00	3.00
Interseção linhas	1.00	2.00	2.00	3.00	3.00	4.00	3.00	3.00	3.00
Média dos 2 ant.	2.00	2.00	2.00	2.00	3.00	3.00	3.00	4.00	3.00

Tabela 5.7: Erro de localização do melhores parâmetros do Filtro de Kalman (em metros)

Erro de localização do melhores parâmetros do Filtro de Kalman (m)									
Alg. \ $ERRO_{RSSI}$	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Distância próx.	3.97	4.82	5.76	6.74	8.07	9.28	10.93	12.65	14.68
Interseção linhas	2.98	3.88	5.12	6.59	8.37	10.20	12.58	14.58	16.88
Média dos 2 ant.	3.16	4.15	5.28	6.46	8.00	9.52	11.55	13.43	15.60

Pela análise das tabelas 5.4 e 5.7 é possível concluir que o filtro de Kalman melhora os valores do erro de localização. A redução do erro é de cerca de 10% em relação aos resultados obtidos pelo filtro de estados do RSSI. Em relação ao comportamento dos algoritmos de localização descrito anteriormente, este continua a verificar-se.

No Anexo B encontram-se as tabelas com os valores de distância percorrida obtidos para as diferentes fases de filtragem. Pela análise destas é possível observar que existe uma enorme redução entre os valores obtidos pelos dados não tratados e os valores obtidos em cada uma das outras fases, sendo estes últimos mais próximos do valor real (1200m). No entanto, esta melhoria nos valores de distância percorrida não se deve aos filtros em si, mas sim ao limitador de distância percorrida que existe no condicionador de localização.

5.3.3 Imagens da localização com dados simulados

Para ser possível fazer uma análise gráfica dos dados foi realizada uma simulação com o parâmetro $ERRO_{RSSI}$ igual a 7. Assim, de acordo com a tabela 5.6 o melhor valor para o parâmetro σ_z usado no filtro de Kalman é igual a 3.00.

Na tabela seguinte estão apresentados os resultados obtidos na simulação.

Tabela 5.8: Resultados da simulação para $ERRO_{RSSI} = 7$ e $\sigma_z = 3.0$.

		Erro (m)	**Redução de erro (%)**	**Distância percorrida (m)**
Distância mais próxima	Original	15.34	0.00	26153.34
	FER	11.15	27.34	2227.91
	FK	8.35	45.58	2181.80
	FE	10.62	30.78	2224.88
Interseção de linhas	Original	16.38	0.00	26850.42
	FER	12.70	22.49	2192.46
	FK	10.21	37.71	2154.72
	FE	12.51	23.65	2162.84
Média do dois anteriores	Original	14.78	0.00	24316.31
	FER	11.82	20.05	1992.86
	FK	9.10	38.40	1940.08
	FE	11.41	22.78	1974.36

Como é possível observar pela tabela 5.8 os melhores erros de localização foram obtidos com o algoritmo da distância mais próxima e, portanto, de seguida são apresentadas as imagens da localização, referentes a esse algoritmo, em cada uma das fases de filtragem. As imagens referentes aos outros algoritmos encontram-se no Anexo C.

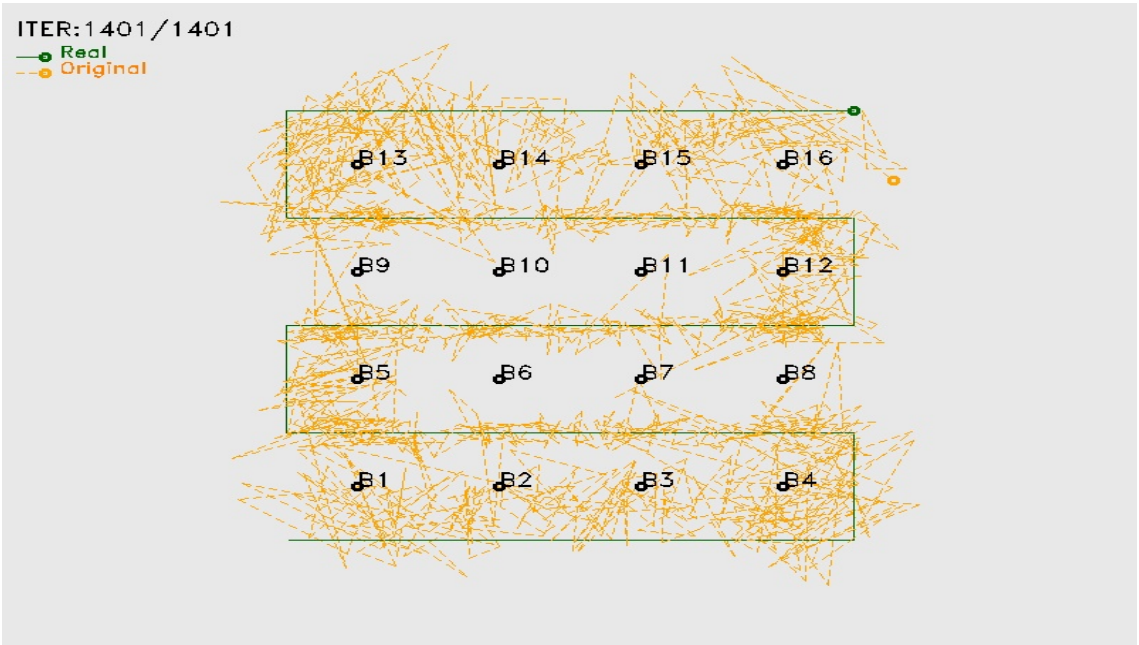

Figura 5.8: Localização dos dados não tratados (algoritmo da distância mais próxima)

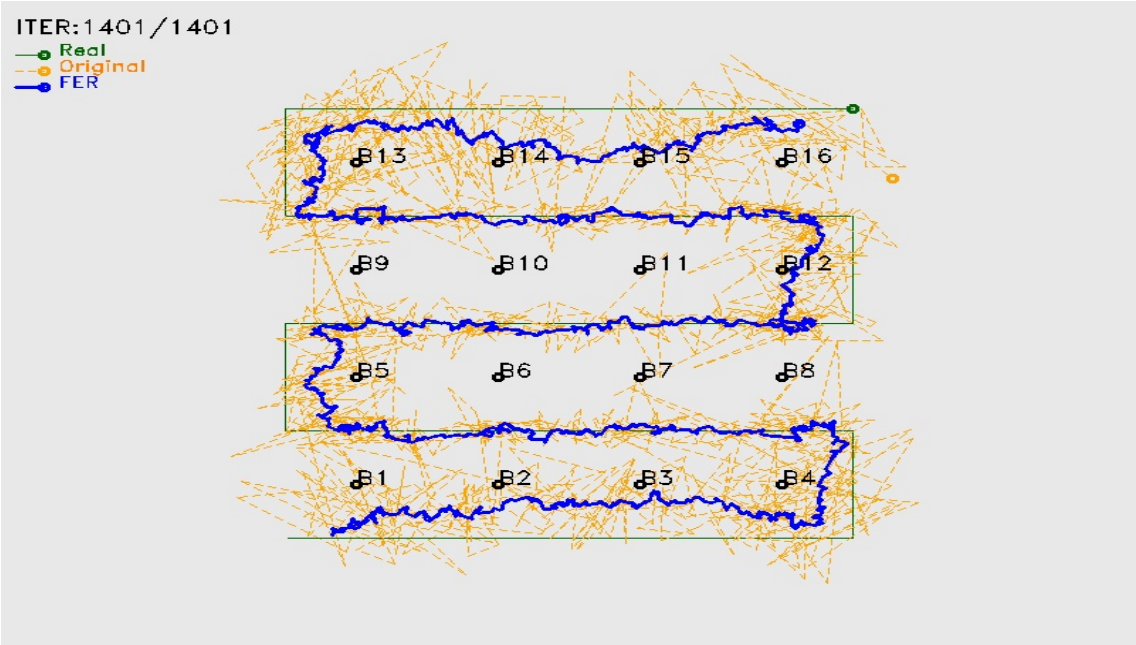

Figura 5.9: Localização dos dados não tratados e dos dados do Filtro de Estados do RSSI (algoritmo da distância mais próxima)

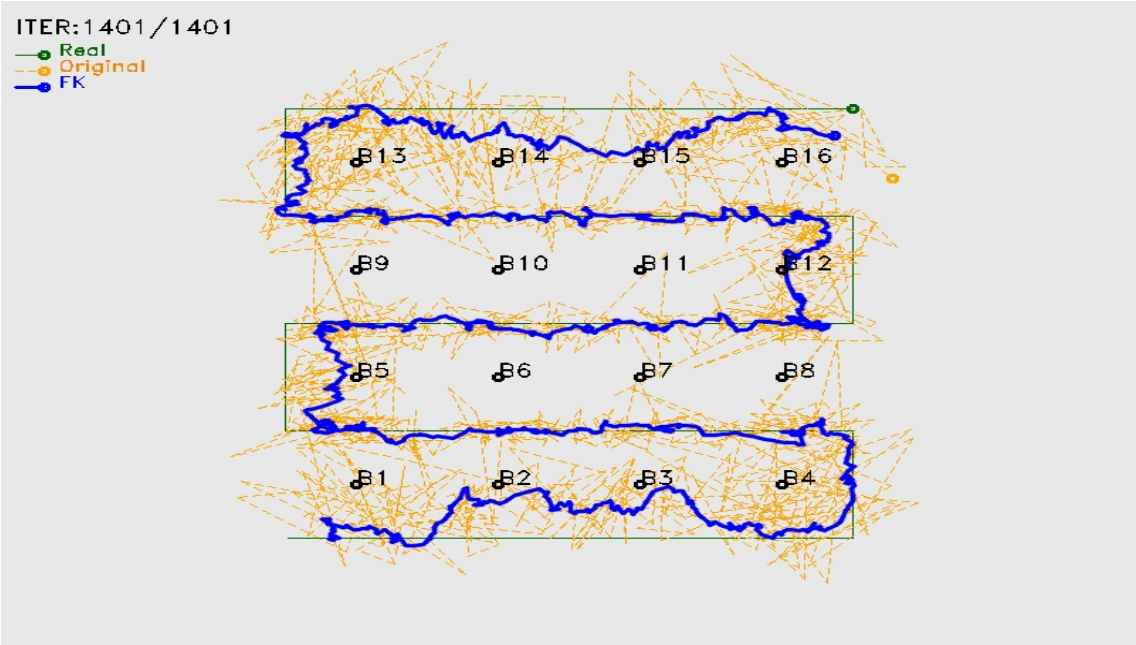

Figura 5.10: Localização dos dados não tratados e dos dados do Filtro de Kalman (algoritmo da distância mais próxima)

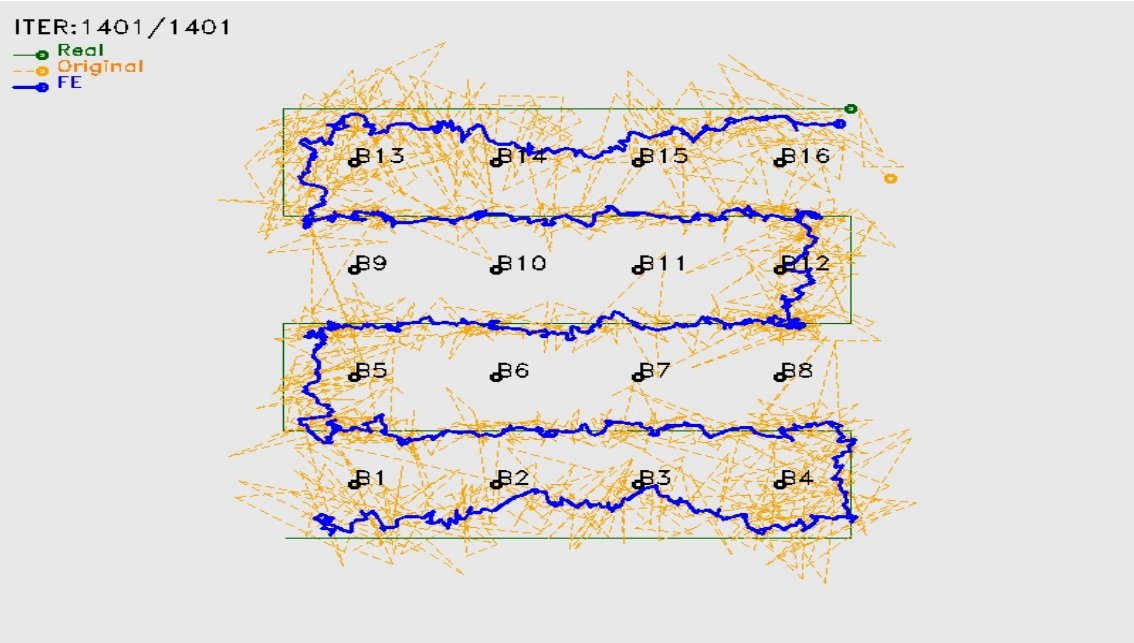

Figura 5.11: Localização dos dados não tratados e dos dados do Filtro de Estados da Distância (algoritmo da distância mais próxima)

Pela análise das imagens é possível verificar que nas zonas interiores, onde as distâncias aos faróis usados na localização são menores, o erro de localização é menor. Por outro lado, nas zonas mais externas do circuito, por exemplo nos troços inicias finais (do farol 1 ao 4, e do farol 13 ao 16) e nos troços mais pequenos do percurso, é possível observar que o erro é maior. Isto deve-se ao facto de as distâncias usadas na localização serem maiores e conterem um erro maior.

Assim é possível concluir que diferença nos erros de localização apresentados na tabela 5.8 é causada maioritariamente pela localização obtida nas zonas exteriores do circuito.

5.3.4 Resultados com dados em ambiente real

Os resultados apresentados a seguir são referentes as simulações efetuadas com dados obtidos num ambiente real. A obtenção destes dados foi feita numa vinha através de uma das coleiras do projeto SheepIT que foi pendurada à cintura de uma pessoa e que foi transportada por entre as filas de videiras. O intervalo de tempo entre duas medições consecutivas foi de 6 segundos. Para cobrir a área da vinha foram posicionados 7 faróis como é possível ver na figura 5.12, sendo que antes destes terem sido posicionados foi executado o processo de calibração a uma distância de 90m.

Para se conseguir estimar erros de localização foram obtidas posições reais através de um módulo GPS contido num Smartwatch [34], com uma precisão de ±19 pés ($\pm 5.79m$). Os valores apresentados referentes ao filtro de Kalman foram obtidos com os parâmetros $\sigma_z = 10.0$ e $\sigma_a = 1.5$.

Figura 5.12: Imagem do Google Maps da Quita do Pato

Tabela 5.9: Resultados da localização com os dados obtidos na Quinta do Pato

		Erro (m)	**Redução de erro (%)**	**Distância percorrida (m)**
Distância mais próxima	Original	77.82	0.00	15607.46
	FER	41.48	46.70	1441.10
	FK	42.12	45.88	1472.88
	FE	43.52	44.07	1288.07
Interseção de linhas	Original	224.31	0.00	73509.05
	FER	39.50	82.39	1862.99
	FK	39.13	82.56	1903.03
	FE	44.12	80.33	1839.66
Média do dois anteriores	Original	138.70	0.00	41020.41
	FER	39.61	71.44	1400.92
	FK	39.40	71.60	1453.46
	FE	40.18	70.03	1269.67

A partir da análise da tabela 5.9 é possível verificar que apesar da redução percentual do erro obtida nas diferentes fases de filtragem ser de 44% a 47% para o algoritmo de distância mais próxima, de 80% a 83% para o algoritmo de interseção de linhas e de 71% para a localização média dos dois algoritmos, o erro obtido com os dados não tratados para cada um dos algoritmos de localização é muitíssimo grande, levando assim a que o erro obtido em cada uma das diferentes fases de filtragem continue a ser grande. Isto deve-se ao facto de em ambiente real poderem existir imensos obstáculos entre os rádios das coleiras e dos faróis, fazendo com que haja reflexões e refrações do sinal de rádio que degradam os valores de RSSI medidos e por sua vez os valores de distância estimados para a localização. Além

disso, existe um outro fator que na altura da recolha dos dados não foi tido em conta, mas que também poderá ser uma das causas de erro, que é relativo ao número de faróis usados para cobrir o terreno e o seu posicionamento. Quantos menos faróis forem usados para cobrir uma determinada área maior será a distância entre eles, e por sua vez maiores serão as distâncias usadas na localização. Isto leva a que, tal como já foi referido anteriormente, o erro cometido na estimativa das distâncias usadas na localização aumente significativamente.

Para se poder fazer uma análise gráfica dos resultados obtidos pelos algoritmos foram adquiridas as imagens resultantes da localização obtida em todas as fases de filtragem para cada um dos algoritmos. De seguida são apresentadas as imagens de localização obtidas com o algoritmo de distância mais próxima. As restantes imagens encontram-se no Anexo D.

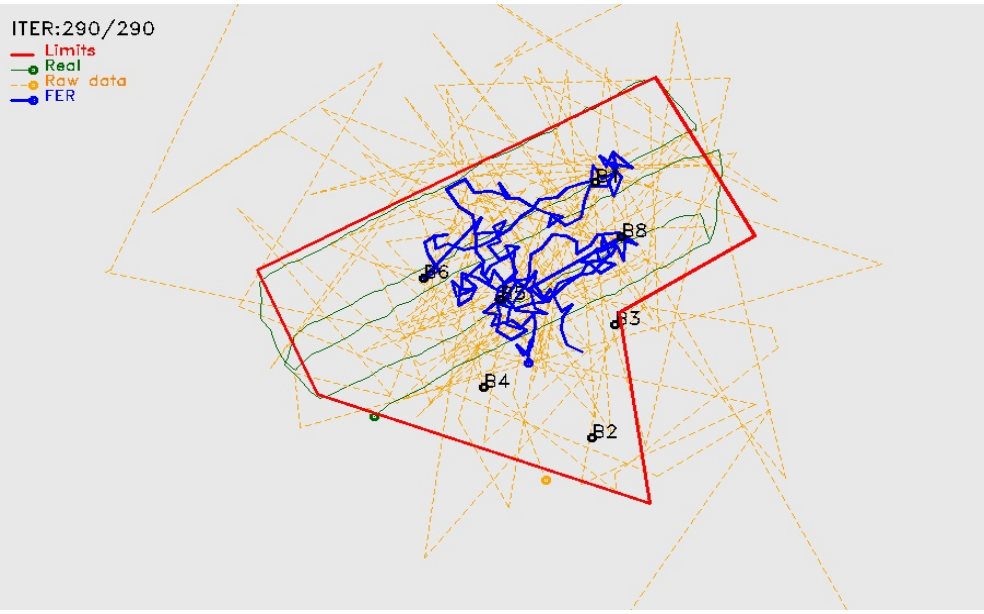

Figura 5.13: Localização do Filtro de Estados do RSSI em ambiente real (algoritmo da distância mais próxima)

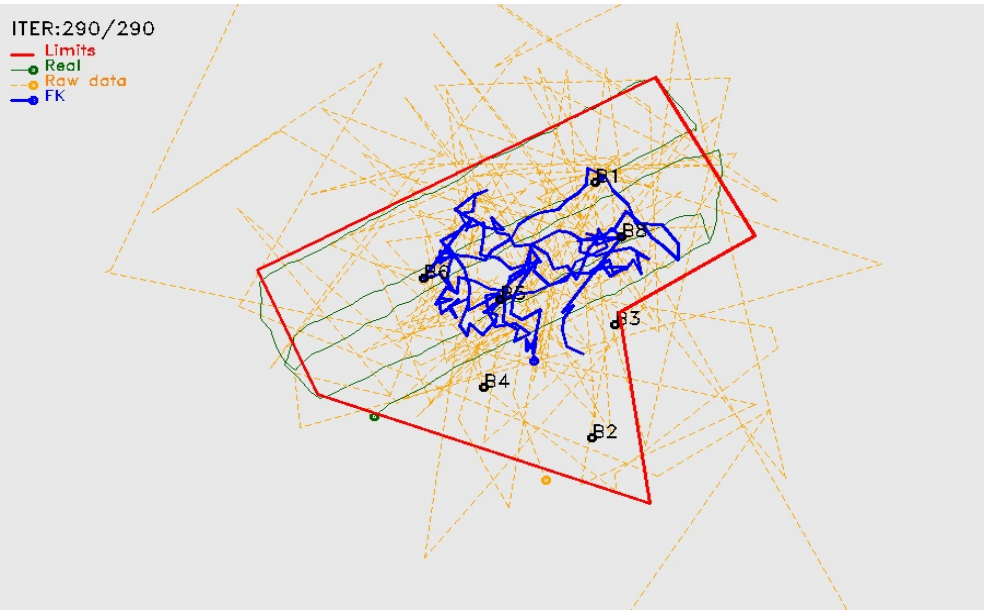

Figura 5.14: Localização do Filtro de Kalman em ambiente real (algoritmo da distância mais próxima)

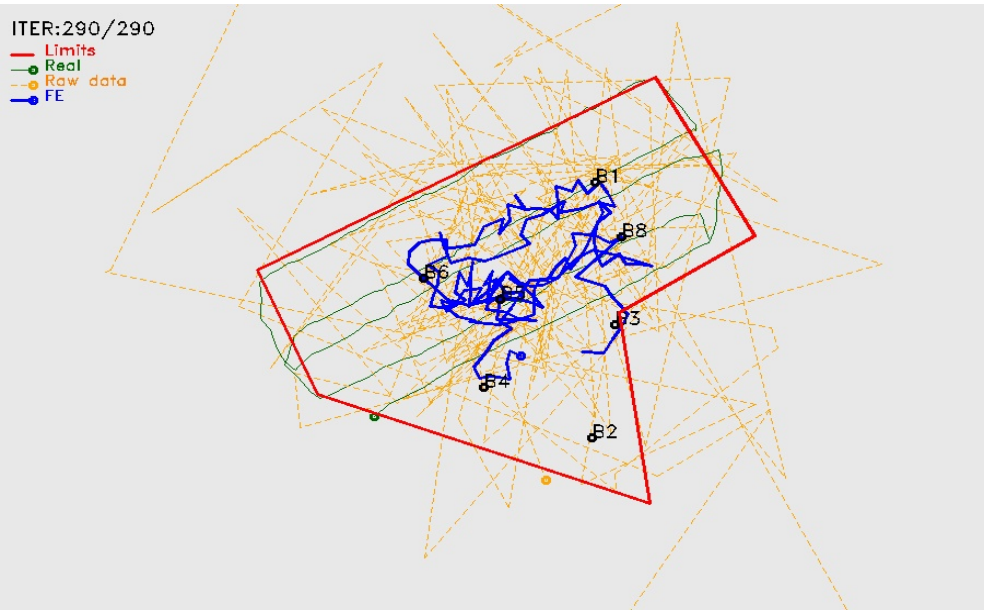

Figura 5.15: Localização do Filtro de Estados da Distância em ambiente real (algoritmo da distância mais próxima)

Pela análise da imagens da localização é possível concluir que nas zonas mais exteriores do percurso a localização obtida é relativamente má, devido as razões enunciadas a cima, e que os valores de erro obtidos na tabela 5.9 são fortemente influenciados por isso.

Assim, para perceber melhor qual é o erro de localização no interior do percurso, foi calculado o erro médio em dois segmentos do percurso, um primeiro que vai do farol 8 até ao

farol 5 e um segundo do farol 6 até ao farol 1 (figura 5.16). Na tabela 5.10 são apresentados os valores de erro obtidos.

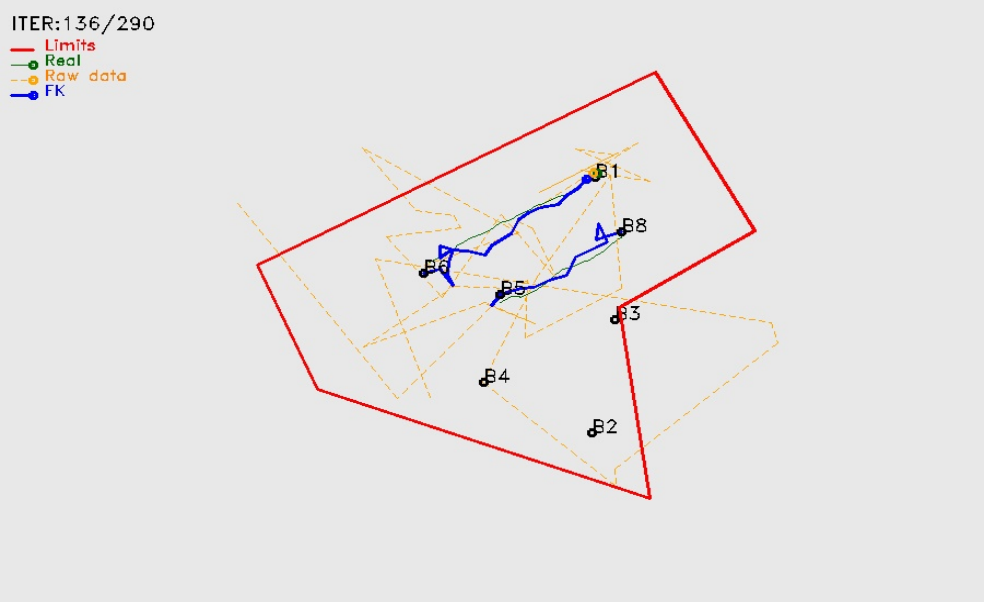

Figura 5.16: Localização dos segmentos com Filtro de Kalaman e o algoritmo de ditância mais próxima

Tabela 5.10: Erros médios de localização dos segmentos do percurso

		Erro (m)
Ditância mais próxima	FER	10.64
	FK	12.95
	FED	17.07
Interseção de linhas	FER	14.39
	FK	14.91
	FED	12.18
Média dos dois anteriores	FER	11.56
	FK	12.75
	FED	13.48

Através da análise da figura 5.16 e da tabela 5.10 é possível concluir que na zona mais interior dos faróis o erro de localização é muito menor, chegando a obter valores de erro 2 a 4 vezes menores em relação aos valores apresentados na tabela 5.9.

5.4 Análise de recursos consumidos

Para avaliar o impacto que os mecanismos desenvolvidos têm na gateway foram efetuados testes onde foram medidos os recursos consumidos pela gateway. Para isso, foram medidos os valores de percentagem de CPU e memória utilizados quando se executava a gateway sem qualquer filtro ativo e depois para um dos filtros de cada vez. As simulações foram feitas num

ambiente Linux através de uma VirtualBox à qual se forneceu 1GB de RAM. Para simular a localização foi criado um dataset para 1000 coleiras usando o sintetizador de dados apresentado na Secção 5.3.1. De modo a conseguir-se perceber a evolução dos recursos necessários com o aumento do número de dispositivos da rede foram feitas simulações para diferentes números de coleiras. Para se tentar igualar as condições de simulação foram usados os dados das N primeiras coleiras do dataset, onde N corresponde ao número de coleiras da simulação. Para cada uma destas simulações foram obtidos vários valores de CPU e memória e foi feita uma estimativa média. O número de dispositivos para os quais foram feitas simulações são os seguintes:

- Faróis: 16;
- Coleiras: 50, 100, 200, 400, 800, 1000.

A obtenção das percentagens de CPU e memória utilizado foi feita através do comando *top -b -n -p $(pgrep -d',' -f gateway)*, onde os parâmetros têm as seguintes funções:

- -b: permite guardar num ficheiro a saída do top;
- -n: permite definir o número de valores a serem guardados pelo top;
- -b: permite monitorizar processos especificados através do seu ID.

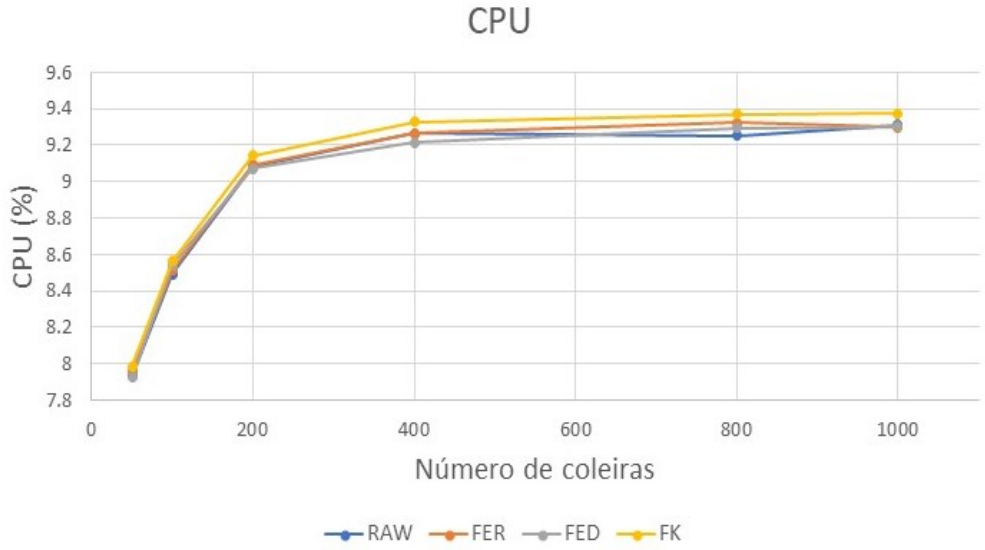

Figura 5.17: Percentagem de CPU utilizado pela gateway

Na figura 5.17 está representado o gráfico referente aos valores de percentagem de CPU utilizado pela gateway. Pela a análise deste é possível constatar que até as 400 coleiras existe um crescimento acentuado na percentagem de CPU utilizada. A partir das 400 coleiras este crescimento estabiliza, não ultrapassando dos 9.4%. É ainda possível verificar que, tal que era esperado, o filtro de Kalman apresenta um maior consumo do CPU comprado com os outros filtros devido à maior complexidade nos cálculos. Comparando os valores obtidos quando não está nenhum filtro ativo (RAW) com os valores obtidos para cada um dos filtros é possível

verificar que os filtros não provocam qualquer tipo de limitação a nivel de percentagem de CPU.

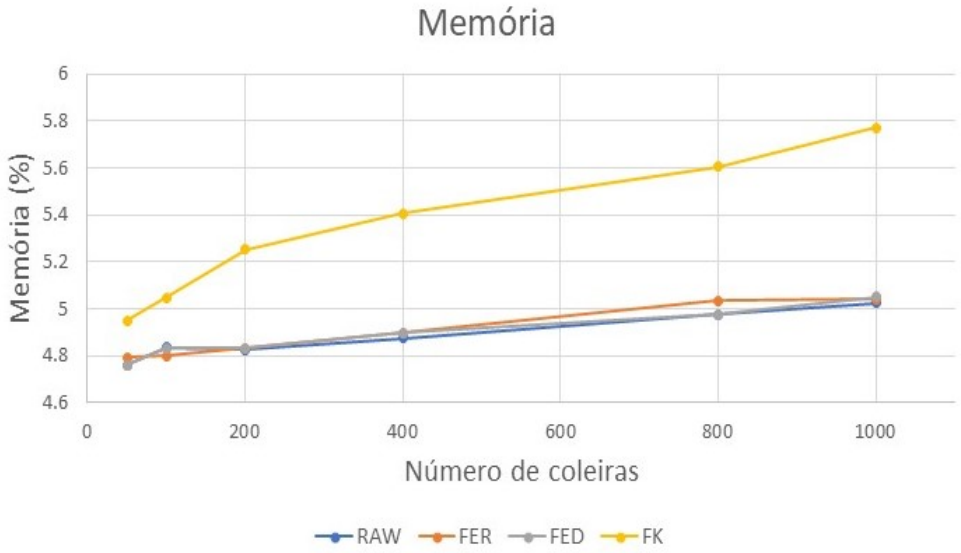

Figura 5.18: Percentagem de memória utilizada pela gateway

Na figura 5.18 está representado o gráfico referente aos valores de percentagem de memória utilizada pela gateway. Analisando este, é possível verificar que o crescimento da percentagem de memória utilizada é linear e pouco acentuado. É possível também observar que a percentagem de memória utilizada pelo filtro de Kalman é superior à utilizada pelos outro filtros. Isto deve-se ao facto de, para alem de ser utilizada uma tabela de NxM, onde N é o número de coleiras e M o número de faróis, de números inteiros para contabilizar as ausências, é também utilizada uma tabela do mesmo tamanho de uma estrutura criada para guardar todas as variáveis de estado, ganhos de Kalman e variáveis das incertezas. A percentagem de memória utilizada pela gateway é máxima de 5.77% para o filtro de Kalman com 1000 coleiras. Esta quando comparada com os 5.02% quando não está nenhum filtro ativo permite concluir que não existem qualquer tipo de limitações a nível de percentagem de memória por parte dos filtros.

5.5 Discussão

Neste capítulo são apresentados todos os procedimentos de teste e resultados obtidos para todos os processos descritos no Capítulo 4. Os testes efetuados ao processo de calibração provaram que este é bastante eficaz na redução da dispersão dos valores de RSSI para diferente dispositivos. Os testes aos mecanismos de filtragem de erro foram efetuados em duas partes. Na primeira parte este foram feitos com dados sintetizados produzindo resultados de localização promissores. Contudo, na segunda parte, quanto estes foram feitos com dados reais, os resultados obtidos já não foram tão bons. Por fim, são apresentados testes e análises a nível de recursos utilizados, nomeadamente percentagens de CPU e memória, tendo sido obtidos bons resultados. Estes resultados levaram às conclusões descritas no Capítulo 6.

Capítulo 6

Conclusões

A delimitação das áreas de pastoreio é um dos principais problemas no sector pecuário. O projeto SheepIT, que visa usar ovinos para a remoção de ervas infestantes nas vinhas, tem como requisito a necessidade de delimitar e controlar as áreas onde estes podem pastar. Para isso é necessário que o projeto SheepIT possua um sistema capaz de localizar os animais em tempo real.

Aquando o início deste trabalho o projeto SheepIT já apresentava uma solução para o sistema de localização de baixo custo. Esta solução passa por obter a localização absoluta dos faróis através de um módulo GPS e por obter a localização relativa das coleiras em relação aos faróis, na gateway, com recurso a algoritmos de trilateração e aos valores de RSSI medidos durante as comunicações entre estes dois tipos de dispositivos.

Nesta dissertação é apresentada a implementação de um sistema de localização baseado na solução já existente, ao qual foram adicionados mecanismos de calibração e filtragem de modo a melhorar as estimativas de distâncias usadas na trilateração. Uma vez que o rádio usado nas comunicações entre as coleiras e os faróis foi alterado, foi necessário obter outro modelo, capaz de relacionar os valores de RSSI com a distância real entre dois dispositivos, onde se verificou que para uma determinada distância real eram estimadas distâncias muito dispares para diferente pares coleira/farol. Para contrariar este problema o modelo foi alterado e foi desenvolvido um processo de calibração para o mesmo. Este processo de calibração foi testado e verificou-se que, com o modelo calibrado, a disparidade dos valores de distância estimados se reduziu significativamente.

Posteriormente, foram desenvolvidos três mecanismos de filtragem de erro, sendo que dois funcionam de acordo com os estados obtidos, um aplicado ao RSSI e outro as distâncias estimadas, e o terceiro que corresponde a um filtro de Kalman aplicado também as distâncias estimadas.

De forma a avaliar os mecanismos desenvolvidos, foram realizados testes onde foi obtida a localização de uma coleira inicialmente com dados sintetizados e posteriormente com dados reais. Os resultados obtidos nos testes com dados sintetizados foram aceitáveis. Quando aplicado aos valores de RSSI um erro aleatório de ± 3.5 unidades de RSSI ($ERRO_{RSSI} = 7$) foram obtidos erros médios de localização de aproximadamente 10m, tendo estes sido mínimos para o Filtro de Kalman e máximos para o Filtro de Estados do RSSI. Dos dois algoritmos de trilateração usados foram obtidos melhores resultados para o algoritmo da distância mais próxima. Foi possível também observar através das imagens da localização que, como já era esperado, a localização é melhor quando a coleira se encontra nas zonas interiores da área

coberta pelos faróis e que quando esta se encontra nas extremidades a localização é degradada, pois a incerteza associada aos valores de RSSI provoca um maior erro nas distâncias estimadas.

Por outro lado, os resultados referentes aos dados em ambiente real não foram tão bons. Os valores de erro médio de localização obtidos nestes testes foram de aproximadamente 40m, sendo que estes foram mínimos para o Filtro de Kalman e máximos para o Filtro de Estados da Distância. Em relação aos algoritmos de trilateração usados, foram obtidos melhores valores de erro usando o algoritmo de interseção de linhas. Tal como nos testes com dados sintetizados, aqui também foi possível observar que o erro de localização quando a coleira se encontra nas zonas mais externas da área coberta pelos faróis é muito maior do que quando esta se encontra no interior, fazendo com que o valor de erro de 40m seja fortemente influenciado pelo erro de localização obtido nas zonas mais externas. Assim, foi feita uma análise do erro em dois segmentos do percurso que passam no interior da área de cobertura e foram obtidos erros compreendidos entre os 10m e os 17m.

Por fim, foram efetuados testes onde foram medidos e analisados os recursos utilizados, a nível de CPU e de memória, pela gateway quando esta não tem nenhum mecanismo de filtragem ativos e quando cada um deles é ativado individualmente. Perante os resultados obtidos foi possível verificar que o filtro de Kalman, como já era de esperar, proporciona um maior consumo de recursos quando comparado com os outros mecanismos. No entanto, comparando os valores obtidos com os filtros ativos com os valores obtidos sem filtros é possível concluir que o impacto dos filtros no consumo de recursos da gateway é mínimo, não havendo quaisquer limitações a nível de CPU e de memória.

Posto isto, é possível concluir que apesar de os mecanismos implementados melhorarem o erro da localização, esta terá sempre um erro muito grande associado. Quando for feita a localização com as ovelhas, principalmente quando estas estiverem juntas ou passarem por baixo das videiras, os valores de erro de localização poderão aumentar, pois existirão mais obstáculos entre as coleiras e os faróis, degradando os valores de RSSI. Contudo, estes valores de erro poderão ser minimizados se forem usados mais faróis para cobrir uma determinada área. Nos testes em ambiente real o posicionamento dos faróis foi feito apenas com o objetivo de ser possível comunicar em todo o terreno, levando a que nas extremidades do terreno as coleiras tenham apenas um ou dois faróis perto. Se este posicionamento for feito de forma a que as coleiras se encontrem sempre entre três faróis, ou seja, se forem colocado faróis nas extremidades do terreno e se no interior forem colocados de forma homogéneas, então será possível melhorar a localização do sistema pois as distâncias estimadas para essa localização terão um menor erro.

6.1 Trabalho Futuro

O trabalho desenvolvido nesta dissertação cumpre com a maioria dos objetivos propostos. Contudo, apesar deste trabalho corresponder a um segundo protótipo de um sistema de localização a ser integrado no projeto SheepIT, existem alguns pontos que podem ser melhorados.

Durante os testes efetuados em ambiente real foi possível concluir que a localização é melhor quando uma coleira se encontra numa zona onde a densidade de faróis por área é maior. Assim, propõe-se que sejam feitos testes, em ambiente real, variando o número de faróis por área e analisando o impacto que isso tenha na localização de modo a obter-se um número ótimo de faróis por metro quadrado.

É também proposto que seja feita uma localização coletiva. Durante os testes no terreno foi

possível reparar que as ovelhas são animais gregários, isto é, a movimentação destas é, em regra, feita em grupo e que muito raramente estas se encontram sozinhas. Neste sentido propõe-se que seja utilizada a localização das várias coleiras para minimizar o erro de localização. A ideia passa por tentar identificar o grupo ou os grupos de animais e comparar as posições de cada um destes animais com os centros dos grupos identificados de modo a se encontrar *outliers* e corrigi-los. Isto poderá servir também para tentar estimar um valor de confiança da posição estimada e um sentido de movimento que pode ser usado para melhorar a estimativa das posições.

Bibliografia

[1] *SheepIT Project,* `http://www.av.it.pt/sheepit/index.html`

[2] L. Nóbrega, P. Gonçalves, P. Pedreiras, e S. Silva, "Energy efficient design of a pasture sensor network," *The 5th International Conference on Future Internet of Things and Cloud- FiCloud 2017,* 2017.

[3] L. Nóbrega, P. Gonçalves, P. Pedreiras, R. Morais e A. Temprilho "SheepIT: Automated Vineyard Weeding Control System" *, em INFORUM simpósio de informática*, 2017.

[4] R. Morais, "Controlo de postura animal" *, Dissertação de Mestrado Integrado em Engenharia Electrónica e Telecomunicações - Departamento de Electrónica, Telecomunicações e Informática - Universidade de Aveiro*, 2017.

[5] J. Pereira, "Sistema de localização de baixo consumo para ovelhas" *, Dissertação de Mestrado em Engenharia de Automação Industrial - Departamento de Engenharia Mecânica - Universidade de Aveiro,* 2018.

[6] *BN-880*, `https://www.hobbyrc.co.uk/beitian-bn-880-gps-compass-module`.

[7] *RabbitMQ*, `https://www.rabbitmq.com/`.

[8] A. Temperilho, "Pastor Virtual" *, Dissertação de Mestrado Integrado em Engenharia Electrónica e Telecomunicações - Departamento de Electrónica, Telecomunicações e Informática - Universidade de Aveiro*, 2017.

[9] A. Cardoso, "Advanced Sheep Posture Controller" *, Dissertação de Mestrado Integrado em Engenharia Electrónica e Telecomunicações - Departamento de Electrónica, Telecomunicações e Informática - Universidade de Aveiro*, 2018.

[10] O. Oguejiofor, V. Okorogu, A. Adewale, B. Osuesu, "Outdoor Localization System Using RSSI Measurement of Wireless Sensor Network" *International Journal of Innovative Technology and Exploring Engineering,* vol. 2, Jan.2013, pp. 1-6.

[11] J. Graefenstein and M. E. Bouzouraa, "Robust Method for Outdoor Localization of a Mobile Robot Using Received Signal Strength in Low Power Wireless Networks," *Proc. International Conference on Robotics and Automation 2008,* 2008.

[12] W. H. Kuo, Y. S. Chen, G. T. Jen, and T. W. Lu, "An intelligent positioning approach: RSSI-based indoor and outdoor localization scheme in Zigbee networks," *Proc. International Conference on Machine Learning and Cybernetics,* vol. 6, 2010, pp. 2754-2759.

[13] B. Bengherbia, M. Ould-Zmirli, S. T. Kebir, H. Hentabli, “"Experimental analysis of RSSI-based outdoor localization in wireless sensor networks, ” *Jurnal Teknologi* 73.2 (2015).

[14] G. Terrasson, A. Llaria, A. Marra, and S. Voaden, “Accelerometer based solution for precision livestock farming: geolocation enhancement and animal activity identification,” *in IOP Conference Series: Materials Science and Engineering,* vol. 138, p. 12004.

[15] A. Llaria, G. Terrasson, H. Arregui, e A. Hacala “Geolocation and monitoring platform for extensive farming in mountain pastures” *A. Llaria, G. Terrasson, H. Arregui, and A. Hacala,* pp. 2420-2425, 2015.

[16] Newman, P., Ward, N., Campbell, H., Watts, M., Franklin, C., and Hunter, J., “OzTrack: Data Management and Analytics Tools for Australian Animal Tracking” *eResearch Australasia, Melbourne.*

[17] J. Hunter, C. Brooking, W. Brimblecombe, “OzTrack-e-Infrastructure to support the management, analysis and sharing of animal tracking data,” *IEEE 9th Int. Conf. on eScience,* pp. 140-147, 2013.

[18] *digitanimal,* `https://digitanimal.com`.

[19] Maroto-Molina, F.; Navarro-García, J.; Príncipe-Aguirre, K.; Gómez-Maqueda, I.; Guerrero-Ginel, J.E.; Garrido-Varo, A.; Pérez-Marín, D.C. “A Low-Cost IoT-Based System to Monitor the Location of a Whole Herd,” Sensors 2019, 19, 2298.

[20] *eShepherd,* `https://www.agersens.com/eshepherd/`

[21] *Nofence,* `https://nofence.no/en/`

[22] A. Srinivasan, J. Wu “A survey on secure localization in wireless sensor networks” *Encyclopedia of Wirelessand Mobile Communications,* Furht B (ed). CRCPress, Taylor and Francis Group: Florida, USA, 2007.

[23] P. Rong and M. Sichitiu, “Angle of arrival localization for wireless sensor networks” *Proc. 3rd Ann. IEEE Commun. Soc. Sens. Ad Hoc Commun. Netw. (SECON'06),* vol. 1, pp. 374-382, 2006.

[24] Y. Chan, W. Tsui, H. So, and P. Ching, “Time-of-arrival based localization under NLOS conditions,” *IEEE Transactions on Vehicular Technology,* vol. 55, no. 1, pp. 17-24, 2006.

[25] H. Kloeden, D. Schwarz, E. B iebl and R. Rasshofer, “Vehicle localization using cooperative RF-based landmarks” *, Proc. IEEE Intelligent Vehicles Symp.,* pp. 387-392, 2011.

[26] F. Gustafsson and F. Gunnarsson ““Positioning using time-difference of arrival measurements,” *in Proc. IEEE Conf. Acoustics, Speech and Signal Processing (ICASSP),* Hong Kong, China, Apr. 2003, vol. 6.

[27] J. Xu, W. Liu, F. Lang, Y. Zhang e C. Wang “Distance Measurement Model Based on RSSI in WSN” *Wireless Sensor Network,* vol. 02, nº 08, pp. 606–611, 2010.

[28] S. Pradhan e S. S. Hwang "Mathematical analysis of line intersection algorithm for TOA trilateration method," *in 2014 Joint 7th International Conference on Soft Computing and Intelligent Systems, SCIS 2014 and 15th International Symposium on Advanced Intelligent Systems, ISIS 2014,* 2014, pp. 1219–1223.

[29] H. Liu, H. Darabi, P.Banerjee, e J. Liu "Survey of Wireless Indoor Positioning Techniques and Systems," *IEEE Trans. Syst. Man Cybern. C: Applications and Reviews*, vol. 37, no. 6, pp. 1067–1080, Nov. 2007.

[30] *Datasheet CC1110,* `https://www.alldatasheet.com/datasheet-pdf/pdf/176864/TI/CC1110.html`

[31] *Datasheet RFM22B,* `https://www.sparkfun.com/datasheets/Wireless/General/RFM22B.pdf`

[32] R. E. Kalman. "A New Approach for to Linear Filtering and Prediction Problems" *Transactions ASME Journal of Basic Engineering,* 1960

[33] G. Welch and G. Bishop, "An introduction to the Kalman filter," Dept. Comput. Sci., Univ. North Carolina, Chapel Hill, Tech. Rep. TR95041, 2000.

[34] Smartwatch: `https://buy.garmin.com/pt-PT/ES/p/83274`

Anexo A

Erros de localização com dados sintetizados

Tabela A.1: Erro médio da localização dos Filtro de Kalman com o algoritmo de distância mais próxima (em metros)

Distância mais próxima									
$\sigma_z \backslash ERRO_{RSSI}$	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
0.05	4.14	5.13	6.26	7.63	9.26	10.92	13.14	15.17	17.30
0.10	4.09	5.03	6.11	7.44	9.03	10.62	12.72	14.79	16.92
0.15	4.04	4.96	6.04	7.27	8.86	10.43	12.44	14.48	16.57
0.20	4.05	4.93	5.96	7.16	8.71	10.27	12.26	14.29	16.30
0.30	4.04	4.90	5.88	7.05	8.53	10.03	11.96	13.94	15.95
0.40	3.99	4.84	5.84	6.97	8.39	9.89	11.77	13.72	15.74
0.50	4.00	4.86	5.79	6.89	8.30	9.76	11.60	13.54	15.54
0.60	4.01	4.83	5.77	6.84	8.23	9.67	11.48	13.44	15.41
0.70	3.97	4.82	5.77	6.81	8.21	9.60	11.41	13.29	15.32
0.80	3.99	4.82	5.76	6.78	8.16	9.57	11.34	13.22	15.24
0.90	3.98	4.82	5.78	6.78	8.16	9.50	11.27	13.12	15.16
1.00	3.97	4.82	5.77	6.77	8.14	9.50	11.20	13.08	15.10
2.00	3.97	4.90	5.88	6.74	8.07	9.32	10.97	12.73	14.78
3.00	4.10	5.01	5.96	6.81	8.09	9.28	10.93	12.69	14.68
4.00	4.25	5.11	6.05	6.89	8.15	9.30	10.94	12.65	14.70
5.00	4.37	5.24	6.15	6.99	8.21	9.33	10.96	12.68	14.76
6.00	4.49	5.38	6.24	7.10	8.25	9.41	11.05	12.74	14.80
7.00	4.67	5.52	6.35	7.20	8.36	9.49	11.11	12.85	14.85
8.00	4.82	5.62	6.45	7.27	8.43	9.55	11.17	12.94	14.92
9.00	4.99	5.75	6.57	7.35	8.54	9.63	11.25	13.02	15.01

Tabela A.2: Erro médio da localização dos Filtro de Kalman com o algoritmo de interseção de linhas (em metros)

Interseção de Linhas									
$\sigma_z \setminus ERRO_{RSSI}$	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
0.05	3.35	4.63	6.14	7.99	9.96	12.02	14.43	16.59	18.92
0.10	3.26	4.49	5.97	7.78	9.71	11.74	14.15	16.27	18.60
0.15	3.21	4.40	5.84	7.61	9.53	11.53	13.94	16.04	18.38
0.20	3.36	4.49	5.76	7.27	9.02	10.81	13.03	15.12	17.34
0.30	3.16	4.32	5.74	7.49	9.39	11.36	13.77	15.87	18.20
0.40	3.10	4.21	5.59	7.29	9.18	11.13	13.52	15.60	17.92
0.50	3.07	4.13	5.49	7.15	9.03	10.96	13.35	15.42	17.73
0.60	3.04	4.08	5.42	7.05	8.91	10.84	13.21	15.28	17.59
0.70	3.02	4.04	5.37	6.98	8.82	10.75	13.10	15.17	17.48
0.80	3.00	4.00	5.32	6.91	8.75	10.67	13.02	15.08	17.40
0.90	2.99	3.98	5.28	6.86	8.69	10.60	12.95	15.01	17.32
1.00	2.98	3.96	5.25	6.81	8.65	10.55	12.90	14.94	17.25
2.00	2.98	3.94	5.22	6.78	8.60	10.50	12.85	14.88	17.20
3.00	2.99	3.88	5.12	6.62	8.42	10.27	12.62	14.64	16.94
4.00	3.06	3.93	5.13	6.59	8.37	10.20	12.58	14.58	16.88
5.00	3.16	4.00	5.18	6.62	8.38	10.20	12.60	14.58	16.88
6.00	3.27	4.10	5.26	6.68	8.43	10.23	12.65	14.61	16.92
7.00	3.39	4.21	5.35	6.76	8.49	10.28	12.72	14.66	16.97
8.00	3.53	4.34	5.46	6.85	8.57	10.33	12.79	14.74	17.04
9.00	3.67	4.47	5.57	6.94	8.66	10.39	12.87	14.80	17.11

Tabela A.3: Erro médio da localização dos Filtro de Kalman com o uso dos algoritmos de interseção de linhas e distância mais próxima(em metros)

Média dos 2 algoritmos de localização									
$\sigma_z \setminus ERRO_{RSSI}$	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
0.05	3.48	4.70	6.03	7.64	9.45	11.31	13.63	15.75	17.98
0.10	3.42	4.58	5.87	7.44	9.20	11.01	13.27	15.40	17.62
0.15	3.36	4.49	5.76	7.27	9.02	10.81	13.03	15.12	17.34
0.20	3.33	4.43	5.67	7.15	8.87	10.65	12.85	14.93	17.11
0.30	3.29	4.35	5.56	6.99	8.67	10.40	12.57	14.63	16.79
0.40	3.25	4.28	5.48	6.87	8.52	10.24	12.39	14.42	16.59
0.50	3.23	4.26	5.41	6.78	8.42	10.11	12.24	14.25	16.41
0.60	3.22	4.22	5.37	6.71	8.33	10.02	12.11	14.14	16.29
0.70	3.19	4.19	5.35	6.66	8.29	9.94	12.03	14.03	16.21
0.80	3.19	4.18	5.32	6.61	8.23	9.90	11.96	13.95	16.13
0.90	3.18	4.17	5.31	6.59	8.21	9.83	11.90	13.87	16.05
1.00	3.17	4.15	5.29	6.56	8.17	9.80	11.84	13.81	16.00
2.00	3.16	4.15	5.28	6.46	8.03	9.58	11.59	13.51	15.69
3.00	3.25	4.21	5.31	6.47	8.00	9.52	11.55	13.45	15.60
4.00	3.37	4.28	5.37	6.51	8.03	9.52	11.55	13.43	15.61
5.00	3.47	4.38	5.44	6.58	8.08	9.54	11.59	13.45	15.66
6.00	3.58	4.49	5.52	6.67	8.12	9.60	11.67	13.51	15.70
7.00	3.72	4.61	5.61	6.75	8.21	9.66	11.72	13.59	15.75
8.00	3.85	4.71	5.71	6.82	8.28	9.72	11.79	13.66	15.82
9.00	3.99	4.83	5.81	6.91	8.37	9.78	11.87	13.74	15.90

Anexo B

Resultados de distância percorrida com dados sintetizados

Tabela B.1: Distância percorrida dos dados não tratados (em metros)

Distância percorrida com os dados não tratados (m)									
Alg. $\backslash ERRO_{RSSI}$	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Distância próx.	12944.03	15712.42	18710.29	22273.46	26427.20	30750.71	34251.40	37702.50	40661.96
Interseção linhas	11804.08	16057.56	19962.97	23780.62	27303.82	30551.75	33506.76	36730.22	39368.46
Média dos 2 ant.	10725.55	13955.30	17171.71	20693.62	24459.46	28158.38	31281.54	34386.68	36997.02

Tabela B.2: Distância percorrida do Filtro de Estados do RSSI (em metros)

Distância percorrida do Filtro de Estados do RSSI (m)									
Alg. $\backslash ERRO_{RSSI}$	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Distância próx.	2362.79	2397.36	2402.17	2404.59	2401.18	2396.59	2390.41	2380.52	2369.02
Interseção linhas	2363.23	2378.11	2375.03	2370.96	2362.48	2354.75	2346.93	2336.30	2329.24
Média dos 2 ant.	2071.08	2106.77	2116.05	2123.50	2133.19	2139.21	2140.09	2132.31	2123.28

Tabela B.3: Distância percorrida do Filtro de Estados da Distância (em metros)

Distância percorrida do Filtro de Estados da Distância (m)									
Alg. $\backslash ERRO_{RSSI}$	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Distância próx.	2292.64	2338.57	2355.98	2366.78	2372.41	2376.25	2373.19	2370.63	2366.40
Interseção linhas	2281.27	2309.56	2314.58	2321.26	2323.01	2322.16	2322.85	2317.45	2320.21
Média dos 2 ant.	1995.14	2035.16	2052.04	2064.21	2080.57	2096.95	2100.40	2102.17	2099.57

Tabela B.4: Distância percorrida do Filtro de Kalman (em metros)

Distância mais próxima									
$\sigma_z \backslash ERRO_{RSSI}$	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
0.05	2357.42	2394.29	2397.51	2402.28	2401.10	2395.43	2390.61	2382.70	2371.87
0.10	2352.13	2389.11	2394.55	2400.42	2400.65	2397.63	2391.02	2383.86	2372.36
0.15	2343.45	2382.20	2393.59	2399.84	2400.70	2397.60	2390.81	2385.47	2373.01
0.20	2331.18	2378.89	2390.71	2396.85	2399.42	2397.84	2391.07	2385.05	2374.78
0.30	2314.52	2370.48	2388.20	2395.73	2399.14	2397.21	2391.77	2385.73	2375.52
0.40	2299.85	2363.58	2383.10	2392.69	2397.64	2396.63	2391.50	2385.52	2377.75
0.50	2285.46	2355.41	2376.61	2391.01	2395.07	2396.79	2391.95	2386.21	2378.15
0.60	2276.21	2349.03	2372.92	2387.47	2393.09	2394.77	2392.20	2387.03	2377.20
0.70	2265.06	2342.02	2370.57	2385.22	2390.80	2393.84	2391.62	2385.75	2378.05
0.80	2257.00	2334.99	2365.96	2382.33	2391.45	2392.47	2392.55	2384.95	2378.76
0.90	2247.42	2331.13	2361.71	2378.53	2389.51	2391.45	2391.23	2384.69	2378.77
1.00	2239.61	2327.10	2358.64	2375.57	2388.80	2390.43	2390.86	2385.45	2378.31
2.00	2193.63	2281.97	2332.64	2357.55	2376.17	2381.53	2384.35	2381.25	2376.99
3.00	2173.75	2258.72	2314.26	2344.81	2365.92	2375.39	2379.74	2376.83	2373.19
4.00	2163.45	2246.48	2301.71	2333.13	2356.16	2368.24	2372.68	2371.01	2369.06
5.00	2156.30	2235.72	2294.27	2326.16	2348.79	2362.03	2366.85	2366.25	2364.67
6.00	2152.17	2230.54	2285.53	2319.50	2344.44	2355.80	2363.05	2360.48	2361.23
7.00	2151.08	2225.02	2281.57	2310.44	2338.87	2349.78	2357.60	2355.46	2357.03
8.00	2149.03	2224.10	2275.93	2307.83	2336.63	2345.13	2354.10	2353.41	2353.16
9.00	2149.65	2221.45	2271.10	2303.95	2332.54	2341.26	2350.43	2349.34	2349.73

Tabela B.5: Distância percorrida do Filtro de Kalman com o algoritmos de interseção de linhas (em metros)

Interseção de Linhas									
$\sigma_z \backslash ERRO_{RSSI}$	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
0.05	2355.83	2375.00	2371.38	2370.99	2363.19	2354.75	2346.99	2336.33	2330.41
0.10	2347.72	2370.42	2369.66	2370.35	2362.17	2355.24	2347.96	2337.35	2331.58
0.15	2339.98	2368.00	2367.47	2368.28	2360.40	2356.34	2348.43	2338.24	2332.70
0.20	2331.76	2362.73	2365.47	2366.86	2359.69	2355.62	2348.48	2338.11	2333.11
0.30	2316.23	2355.26	2361.85	2363.82	2357.44	2354.32	2348.15	2338.43	2333.22
0.40	2302.12	2347.48	2356.40	2361.54	2355.79	2353.77	2347.82	2339.01	2332.98
0.50	2288.02	2338.50	2353.21	2359.25	2355.36	2353.46	2347.99	2338.40	2333.48
0.60	2275.63	2330.67	2349.12	2355.94	2353.87	2352.62	2347.57	2339.40	2333.62
0.70	2263.34	2324.11	2343.65	2353.18	2352.26	2351.51	2346.72	2339.48	2333.88
0.80	2251.46	2316.86	2339.60	2351.22	2350.75	2350.54	2346.23	2338.53	2333.97
0.90	2241.63	2310.86	2335.71	2348.10	2349.47	2349.54	2345.99	2337.66	2334.68
1.00	2232.53	2303.24	2332.19	2345.78	2347.84	2348.95	2345.02	2337.62	2333.76
2.00	2155.86	2249.60	2297.10	2324.30	2334.87	2338.76	2340.14	2333.35	2331.75
3.00	2110.69	2211.41	2271.13	2304.81	2319.86	2330.18	2331.37	2329.37	2328.60
4.00	2078.70	2184.90	2247.28	2286.24	2307.97	2319.43	2324.66	2323.17	2324.71
5.00	2056.89	2164.73	2231.22	2271.80	2296.89	2311.64	2318.46	2317.82	2322.23
6.00	2039.45	2149.71	2216.87	2259.10	2286.22	2302.98	2312.47	2313.50	2318.37
7.00	2026.88	2138.42	2206.83	2248.40	2277.51	2294.47	2307.58	2308.94	2314.37
8.00	2018.22	2129.82	2198.80	2238.22	2270.99	2287.39	2303.20	2303.83	2312.21
9.00	2011.39	2122.87	2191.51	2231.25	2265.08	2281.55	2299.38	2298.92	2309.72

Tabela B.6: Distância percorrida do Filtro de Kalman com o uso dos algoritmos de interseção de linhas e distância mais próxima(em metros)

Média dos 2 algoritmos de localização									
$\sigma_z \backslash ERRO_{RSSI}$	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
0.05	2063.38	2106.09	2110.77	2123.32	2133.97	2136.75	2137.59	2133.37	2124.42
0.10	2058.11	2102.48	2108.86	2120.68	2132.26	2136.78	2137.62	2131.77	2123.05
0.15	2051.23	2097.97	2107.33	2118.32	2130.49	2133.86	2138.66	2133.13	2122.40
0.20	2042.33	2091.67	2106.03	2114.41	2129.84	2132.50	2136.26	2131.66	2122.29
0.30	2026.51	2083.80	2102.82	2111.98	2126.58	2130.47	2132.84	2131.04	2122.47
0.40	2014.19	2076.01	2096.87	2109.34	2123.34	2127.66	2131.68	2130.61	2121.59
0.50	2001.25	2066.39	2092.29	2106.18	2121.67	2126.64	2130.48	2128.13	2120.26
0.60	1991.52	2061.92	2088.36	2104.37	2120.41	2124.73	2129.36	2126.54	2118.44
0.70	1980.06	2054.42	2084.84	2100.61	2117.81	2123.97	2129.44	2125.09	2118.68
0.80	1971.79	2047.61	2079.64	2098.47	2115.48	2122.71	2127.25	2123.95	2117.82
0.90	1964.93	2043.74	2075.40	2095.19	2111.91	2120.67	2124.68	2122.67	2116.48
1.00	1957.82	2038.23	2072.09	2092.22	2111.73	2118.76	2124.09	2120.69	2115.74
2.00	1907.45	1992.36	2039.22	2068.64	2095.32	2104.79	2113.52	2113.47	2110.58
3.00	1879.59	1964.16	2018.66	2052.70	2080.90	2096.13	2103.41	2106.92	2103.49
4.00	1863.02	1949.89	2001.97	2036.79	2068.81	2086.83	2095.87	2100.00	2098.77
5.00	1851.99	1935.73	1990.87	2025.48	2057.25	2080.56	2088.26	2093.00	2094.51
6.00	1843.58	1925.44	1980.65	2017.44	2048.97	2072.53	2082.43	2088.55	2091.55
7.00	1836.48	1917.28	1973.75	2008.08	2040.60	2066.37	2076.90	2084.15	2087.11
8.00	1832.24	1912.77	1965.23	2001.86	2035.97	2060.55	2074.03	2080.58	2083.46
9.00	1828.98	1907.38	1958.97	1996.55	2031.03	2055.21	2071.67	2076.05	2080.88

Anexo B

Imagens da localização com dados sintetizados

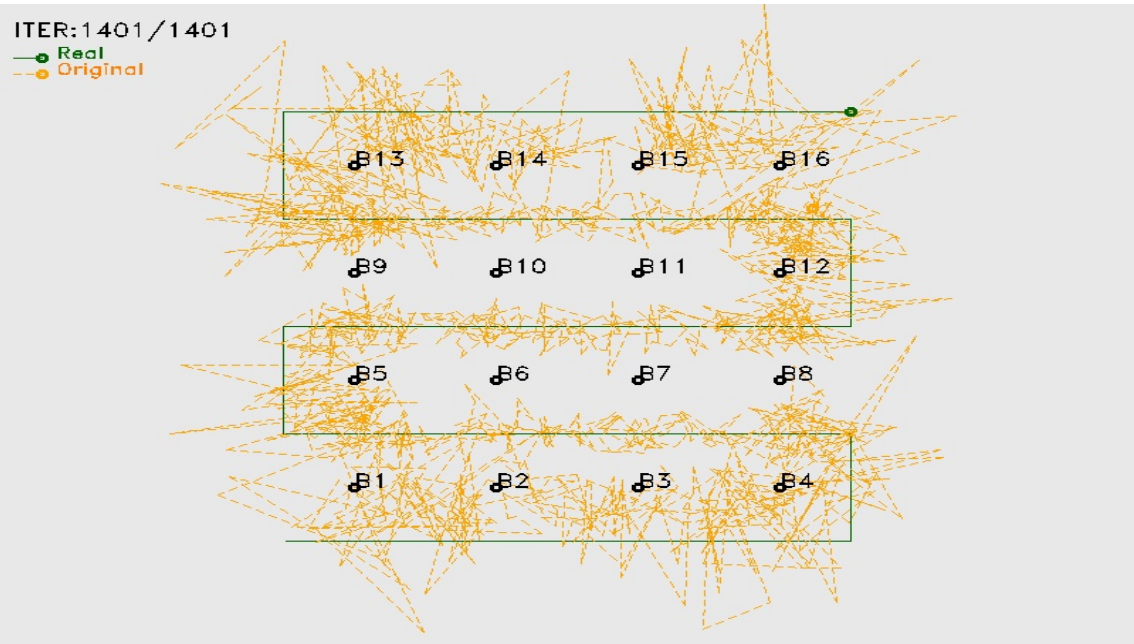

Figura B.1: Localização dos dados não tratados (algoritmo de interseção de linhas)

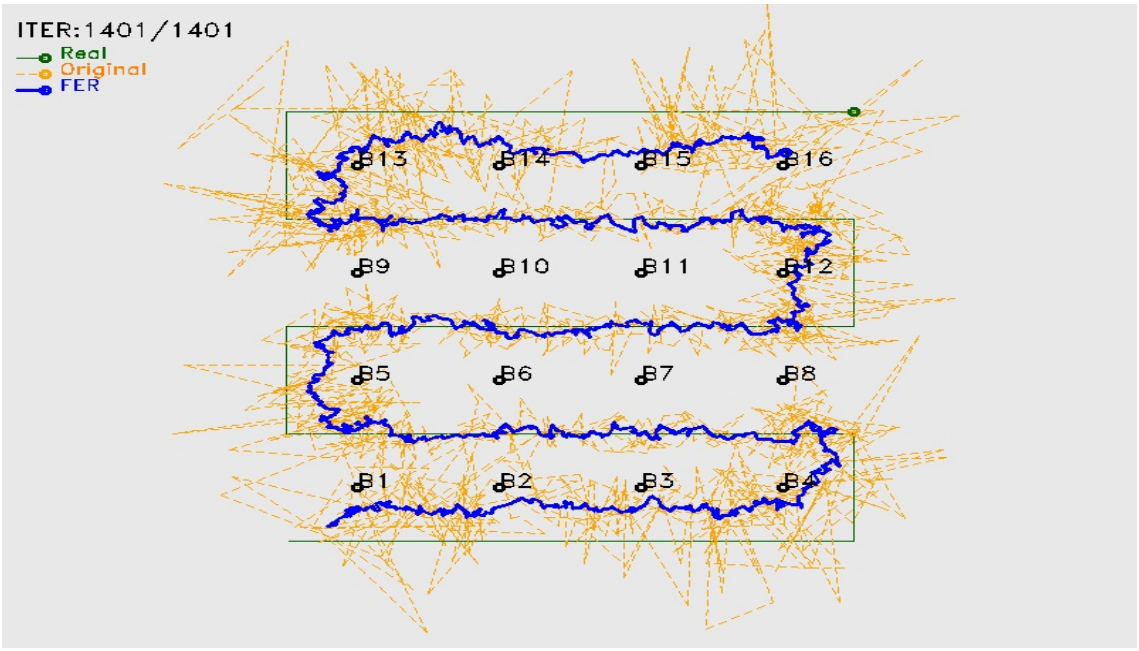

Figura B.2: Localização dos dados não tratados e dos dados do Filtro de Estados do RSSI (algoritmo de interseção de linhas)

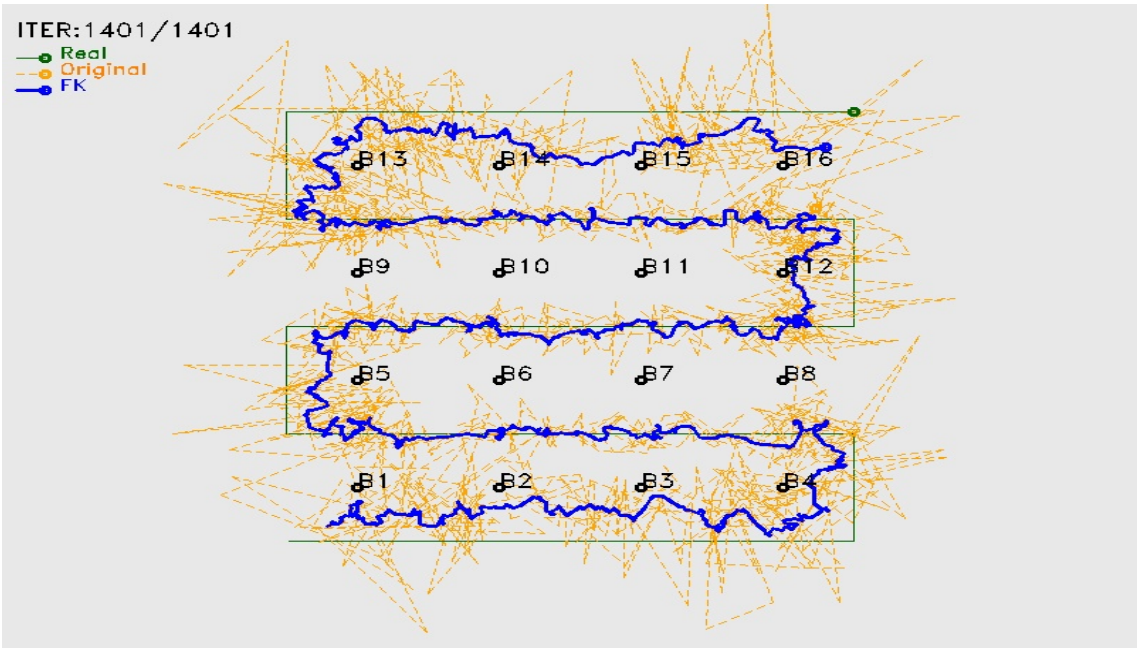

Figura B.3: Localização dos dados não tratados e dos dados do Filtro de Kalman (algoritmo de interseção de linhas)

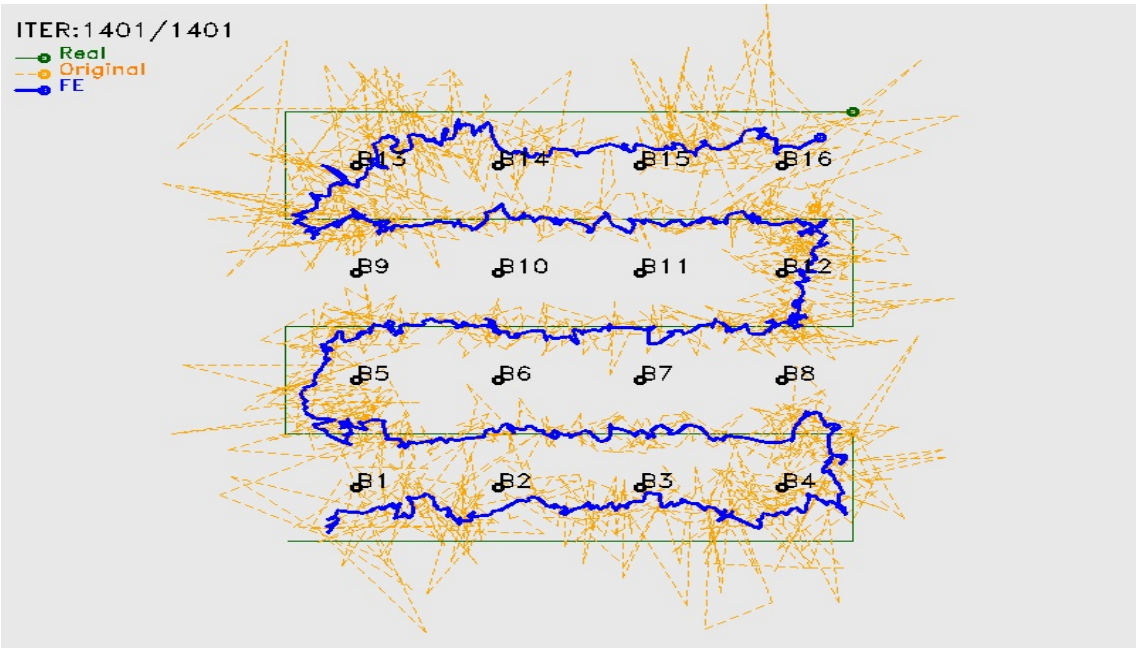

Figura B.4: Localização dos dados não tratados e dos dados do Filtro de Estados da Distância (algoritmo de interseção de linhas)

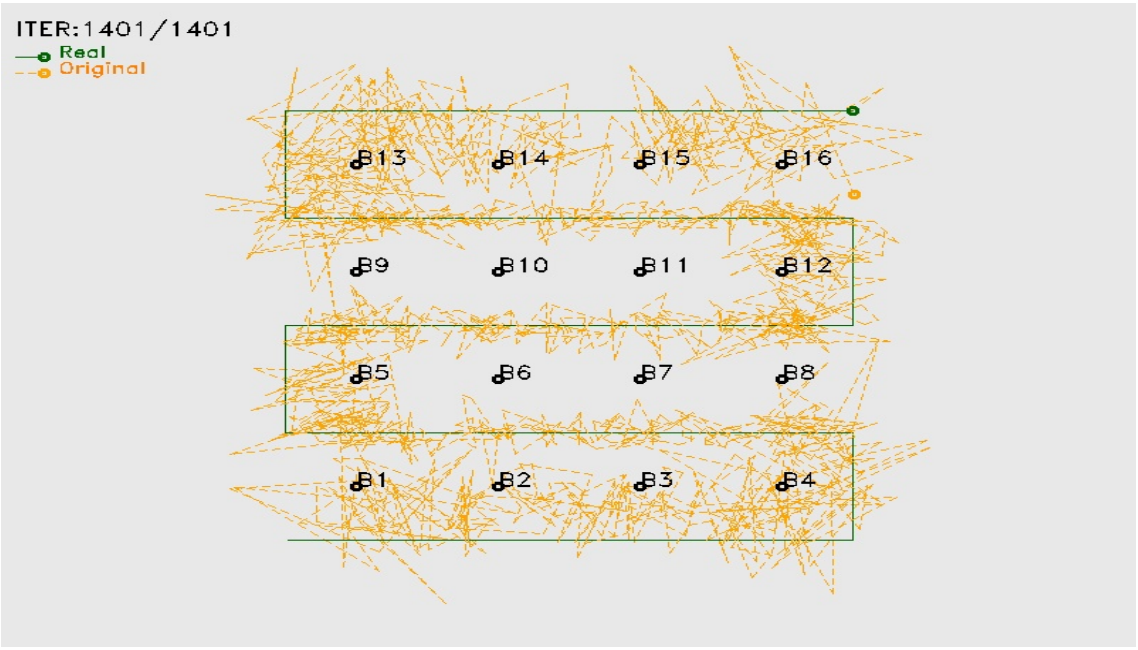

Figura B.5: Localização dos dados não tratados (média do algoritmo de interseção de linhas e distância mais próxima)

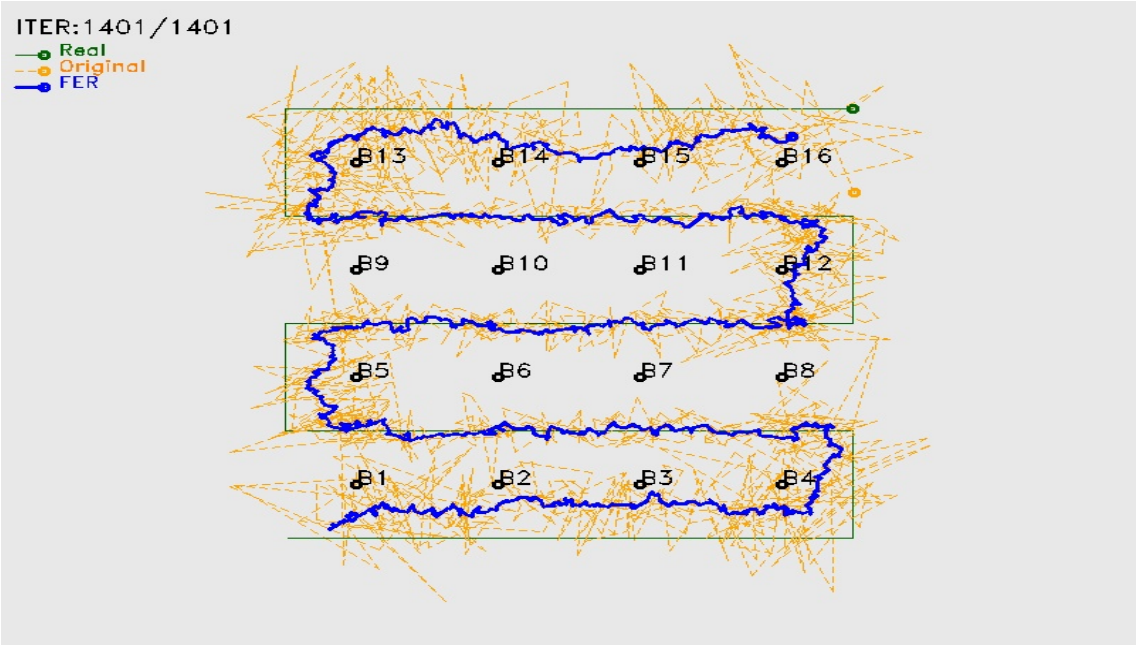

Figura B.6: Localização dos dados não tratados e dos dados do Filtro de Estados do RSSI (média do algoritmo de interseção de linhas e distância mais próxima)

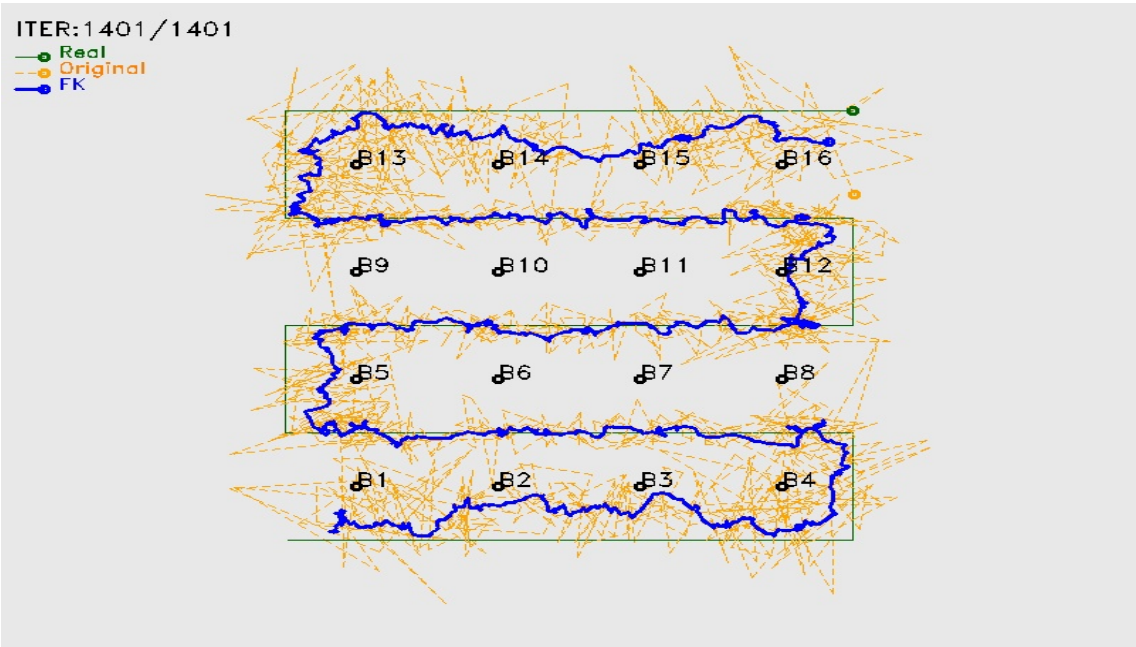

Figura B.7: Localização dos dados não tratados e dos dados do Filtro de Kalman (média do algoritmo de interseção de linhas e distância mais próxima)

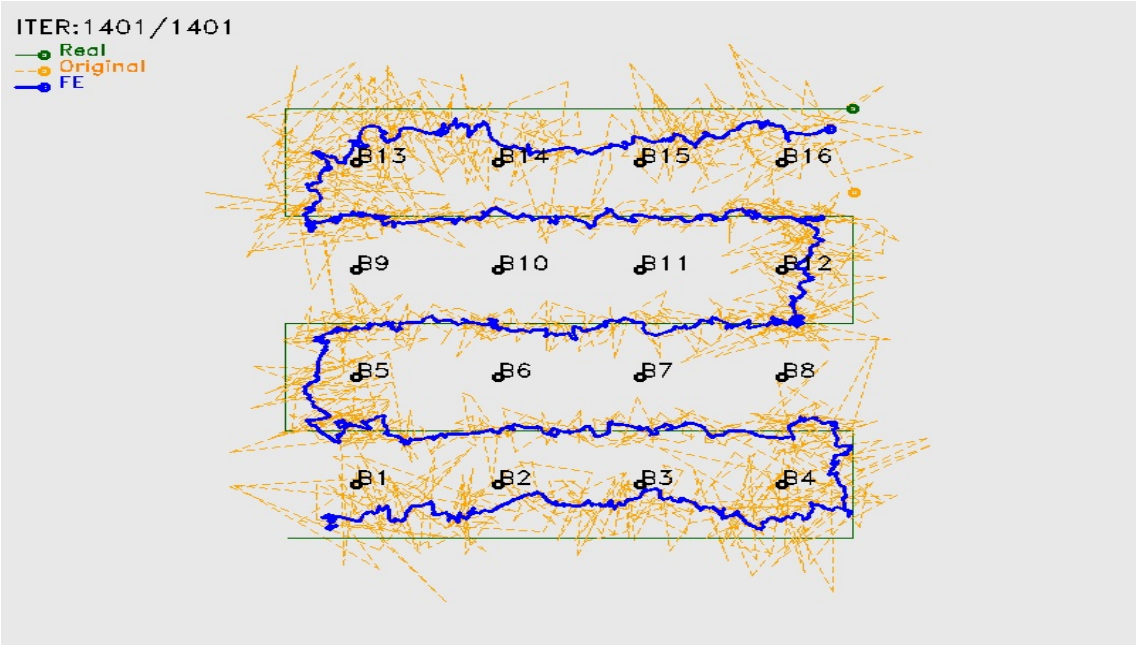

Figura B.8: Localização dos dados não tratados e dos dados do Filtro de Estados da Distância (média do algoritmo de interseção de linhas e distância mais próxima)

Anexo C

Imagens de localização com dados em ambiente real

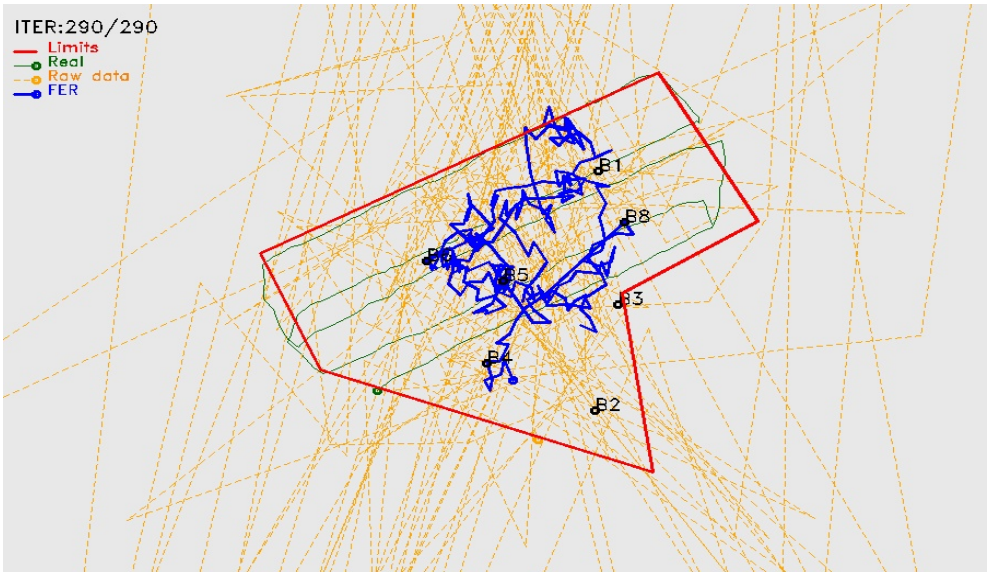

Figura C.1: Localização do Filtro de Estados do RSSI em ambiente real (algoritmo de interseção de linhas)

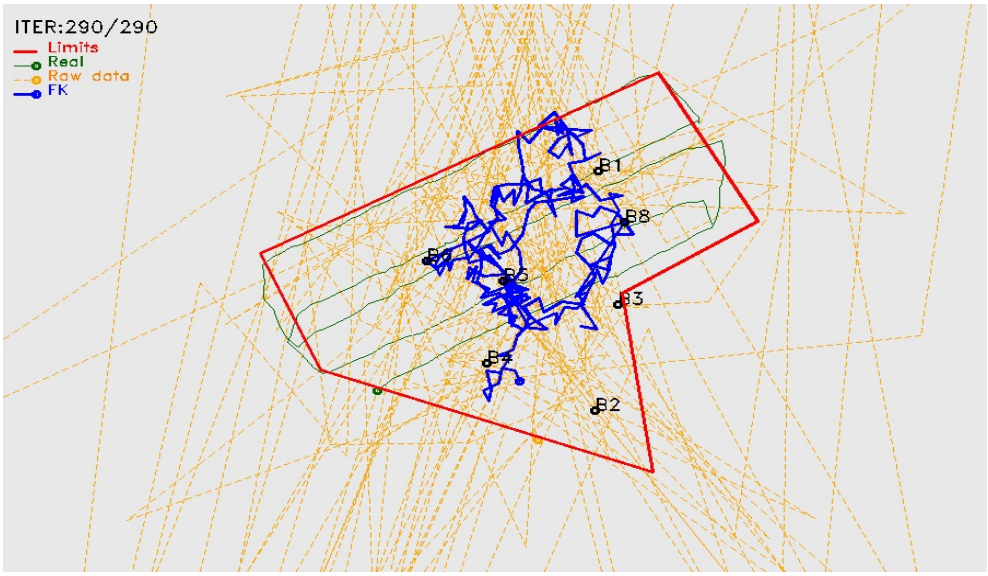

Figura C.2: Localização do Filtro de Kalman em ambiente real (algoritmo de interseção de linhas)

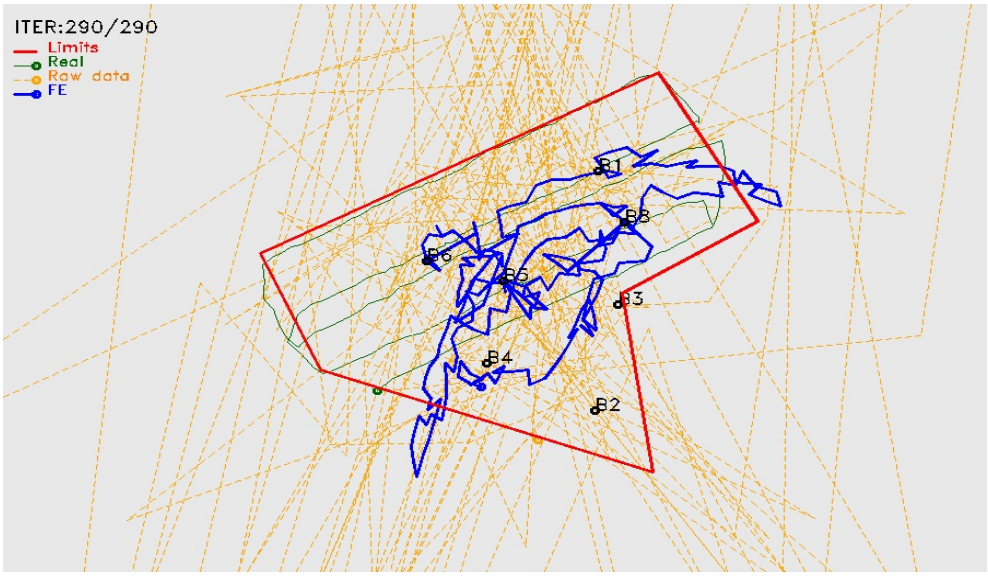

Figura C.3: Localização do Filtro de Estados da Distância em ambiente real (algoritmo de interseção de linhas)

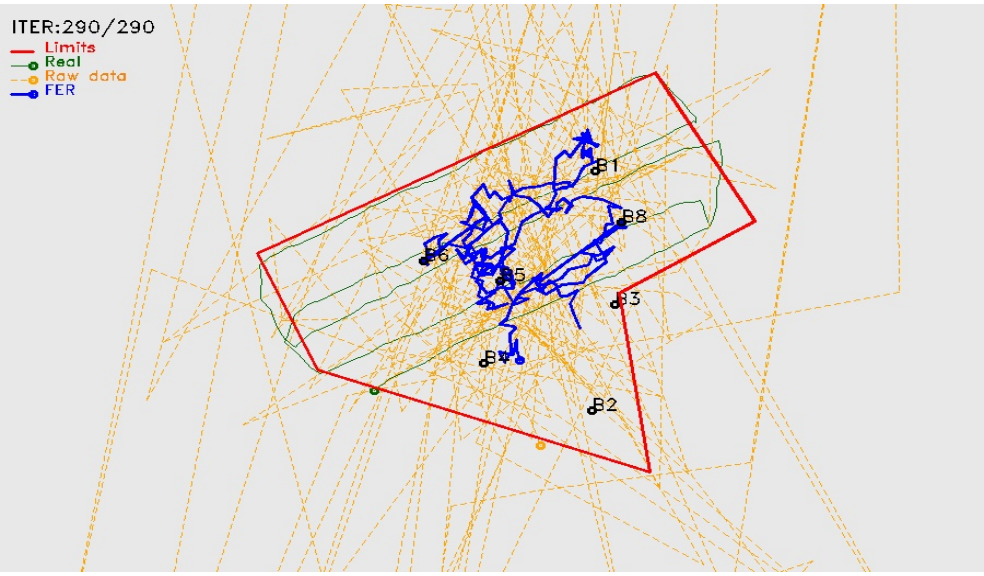

Figura C.4: Localização do Filtro de Estados do RSSI em ambiente real (média do algoritmo de interseção de linhas e distância mais próxima)

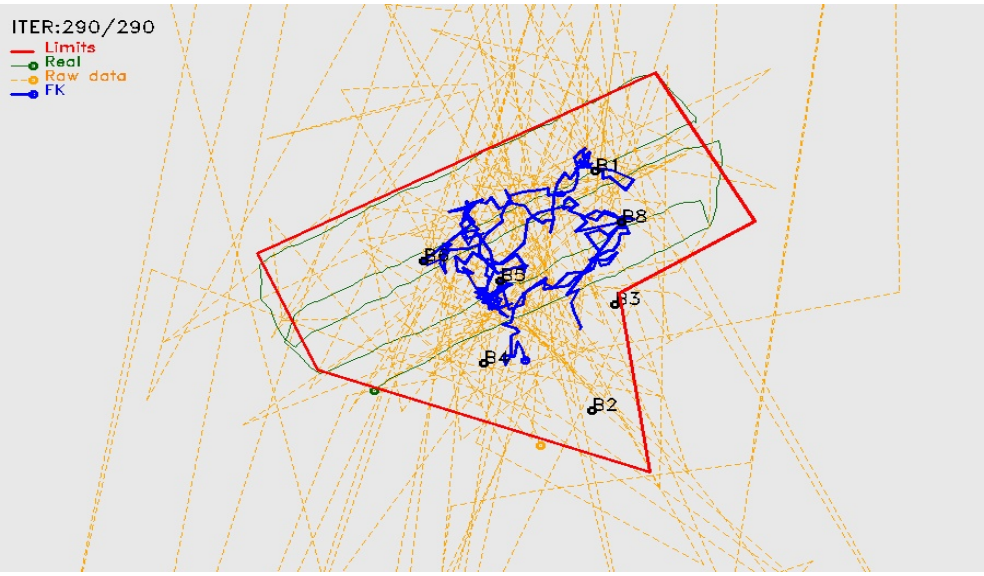

Figura C.5: Localização do Filtro de Kalman em ambiente real (média do algoritmo de interseção de linhas e distância mais próxima)

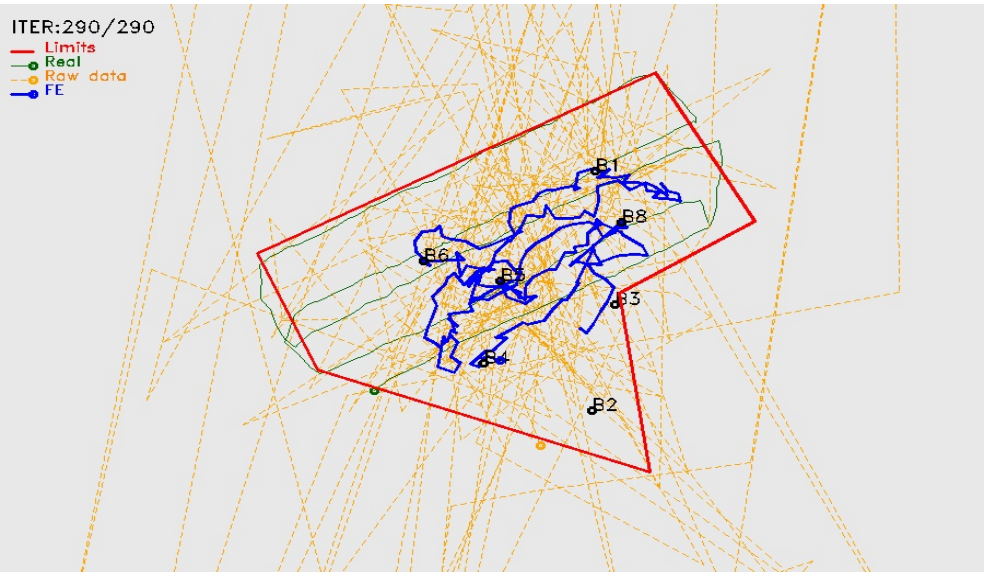

Figura C.6: Localização do Filtro de Estados da Distância em ambiente real (média do algoritmo de interseção de linhas e distância mais próxima)